Forschungszentrum Karlsruhe
Technik und Umwelt
Wissenschaftliche Berichte
FZKA 5872

Modellbildung und Echtzeitsimulation deformierbarer Objekte zur Entwicklung einer interaktiven Trainingsumgebung für die Minimal-Invasive Chirurgie

Ch. Kuhn
Institut für Angewandte Informatik

Januar 1997

Forschungszentrum Karlsruhe
Technik und Umwelt

Wissenschaftliche Berichte
FZKA 5872

Modellbildung und Echtzeitsimulation deformierbarer Objekte zur Entwicklung einer interaktiven Trainingsumgebung für die Minimal-Invasive Chirurgie

Christian Kuhn

Institut für Angewandte Informatik

Als Dissertation genehmigt von der Fakultät für Informatik
der Universität Fridericiana zu Karlsruhe

Forschungszentrum Karlsruhe GmbH, Karlsruhe
1997

Forschungszentrum Karlsruhe GmbH
Postfach 3640, 76021 Karlsruhe

ISSN 0947-8620

Zusammenfassung

Modellbildung und Echtzeitsimulation deformierbarer Objekte zur Entwicklung einer interaktiven Trainingsumgebung für die Minimal-Invasive Chirurgie

Die vorliegende Arbeit beschäftigt sich mit der Entwicklung von Verfahren zur Repräsentation deformierbarer Objekte innerhalb einer interaktiven graphischen Simulationsumgebung. Ziel ist eine möglichst realitätsgetreue dreidimensionale Bewegungs- und Verhaltenssimulation unter Einwirkung äußerer Anregungen.

Wichtige Anforderung für Interaktivität ist die Möglichkeit der direkten Einflußnahme auf das Simulationsszenario und die unmittelbare Reaktion der betroffenen Objekte. Bedingung hierfür ist die Echtzeitfähigkeit der entwickelten Verfahren, worin auch die zentrale Forderung dieser Arbeit liegt. Diese Form der interaktiven Computergraphik wird häufig als 'Virtuelle Realität' bezeichnet, wobei schon der Name den Wunsch nach möglichst guter Nachbildung der Realität impliziert.

Die Simulation der physikalischen Eigenschaften erfordert die Erstellung eines adäquaten mathematischen Modells, das hauptsächlich auf Sachverhalten der Mechanik basiert. Ausgehend von einer allgemeinen Begriffsbestimmung deformierbarer Körper werden die Anforderungen an die Modellbildung diskutiert und auch Parallelen zur 'Finite-Elemente-Methode' (FEM) aufgezeigt. Auf Grundlage von 'nodalen Netzen' wird das MESD-Verfahren (*Methodik der Modellbildung zur echtzeitfähigen Simulation und Manipulation deformierbarer Objekte über physikalisch-basierte nodale Systeme*) erarbeitet. Diese Modelle sind charakterisiert durch die Anwendung physikalischer Grundgleichungen auf diskrete Objektkomponenten, im speziellen nulldimensionale, massebehaftete Knoten und kraftübertragende Verbindungselemente. Das MESD-Verfahren kann als vereinfachte Finite-Elemente-Methode angesehen werden, das auf echtzeitfähige Bewegungssimulation optimiert ist. Hierzu werden Aspekte der numerischen Simulation und der Implementierung auf Rechnersystemen erörtert.

Um eine interaktive Manipulation der Modelle durch den Bediener zu ermöglichen, sind Verfahren zur echtzeitfähigen Kollisionserkennung sowie zur Auswertung der Interaktion entwickelt worden. Insbesondere die Reaktion der deformierbaren Objekte ist in Abhängigkeit der Manipulation zu spezifizieren und zu simulieren. Bei Objektmodifikationen ist eine Strukturänderung der MESD-Modelle auszuführen. Neben den physikalischen Eigenschaften werden hierbei auch regelbasierte Verhaltensmodelle benötigt.

Die graphische Repräsentation besteht in der Visualisierung der Simulationsergebnisse auf einem Bildschirm in Form von geometrischen Modellen der deformierbaren Objekte. Ziel ist eine möglichst realistisches Aussehen. Hier werden zwei Ansätze vorgestellt, ein Ansatz auf Basis von Freiformflächen sowie eine direkte Darstellung von polygonalen Netzen.

Die erarbeiteten Verfahren sind Grundlage für die Entwicklung des 'Karlsruher Endoskopie-Trainers', mit dem die Ausbildung und das Training in der Minimal-Invasiven Chirurgie (MIC) sinnvoll unterstützt werden kann. Das Operationsfeld wird hierbei durch Modelle auf Basis deformierbarer Objekte nachgebildet, wobei der Bediener über eine realistische Bedienerschnittstelle interaktiv Einfluß auf das Szenario nehmen kann. Typische Operationsvorgänge können so in Echtzeit simuliert werden, wobei die Sicht einer Endoskopkamera vom Simulationssystem virtuell erzeugt wird.

Weitere Anwendungsgebiete liegen in den Feldern der interaktiven Computergraphik und Simulationstechnik, in denen die Echtzeitbedingung eine besondere Rolle spielt. Beispiele sind 'Virtual Reality'-Anwendungen, die Simulation technischer Systeme und die schnelle Rekonstruktion von Objekten aus Meßdaten.

Abstract

Modeling and Realtime Simulation of Deformable Objects for the Development of an Interactive Training Environment for Minimally Invasive Surgery

This thesis treats the development of methods for the representation of deformable objects embedded in an interactive graphical simulation environment. Goal is to achieve a highly realistic three-dimensional simulation of object behavior under effect of external stimulations.

An important demand for interactive systems is the possibility of directly influencing the simulation scenario including the immediate reaction of the concerned objects. Therefore the developed methods necessarily have to be realtime capable. This kind of interactive computer graphics is usually called 'Virtual Reality', by which the name implicates the desire of a good imitation of reality.

For simulation of the physical properties, the construction of an equivalent mathematical model is necessary which is based mainly on principles of mechanics. After general definitions, the demands of the modeling procedures are discussed and parallels to the method of Finite-Elements (FEM) are treated. Using the foundation of 'nodal nets', the MESD-procedure (*methods of modeling for realtime capable simulation and manipulation of deformable objects using physically-based nodal systems*) is worked out. This models are characterized by the application of basic physical equations on discrete object components, particularly zero-dimensional mass knots and force-transferring binding elements. The MESD-procedure can be considered as an simplified finite-element method, optimized for realtime capable simulation of body movements. Moreover, aspects of numerical simulation and implementation on computer systems are explained.

To provide an interactive manipulation of the objects by the user, several procedures for realtime collision detection and evaluation of interactions were developed. Especially the reaction of the deformable objects dependent on the manipulation is specified and simulated. Referring to object modifications, it is necessary to change and adapt the structure of the MESD-models. In addition to the physical properties, rule-based behavior models are used which define the object's reaction on different manipulations.

The graphical representation consists of the visualization of the simulation results on a display using geometrical models of the deformable objects. Goal is to achieve a highly realistic visual impression. Two methods are presented, an approach based on free-form surfaces as well as the direct output of polygonal nets.

The procedures resulted of this work are the basis for the development of the 'Karlsruhe Endoscopic Surgery Trainer'. This system can support efficiently the education and training of surgeons in Minimally-Invasive Surgery. The operation area is imitated by models based on deformable objects. The user can interactively manipulate the objects in the scenario by means of a realistic user interface. Herewith, it is possible to execute typical operation procedures. The 'synthetic' view of the endoscopic camera is generated in realtime by the simulation system.

Other application fields exist in such branches of interactive computer graphics and simulation technique where realtime demands play a crucial role. Examples are applications in 'Virtual Reality', the simulation and on-line optimization of mechanical systems and the fast reconstruction of objects using measured data.

Inhaltsverzeichnis

Symbole und Formelzeichen - Übersicht

Mengen werden durch kursive Großbuchstaben gekennzeichnet:

D	Deformierbarer Körper, bestimmt durch Punktemenge
F	Menge der F-Knoten
G	Menge der G-Knoten
E	Menge von Federelementen
K	Menge der Masseknoten
N	Nodales System (Menge der Objektkomponenten)
P	Punktemenge eines allgemeinen Körpers
S	Starrkörper, bestimmt durch Punktemenge
V	Menge der Verbindungselemente
Z	Menge der Z-Knoten

Vektorielle Größen und **Matrizen** werden fett dargestellt, Matrizen stets mit Großbuchstaben:

$\mathbf{D}_\varepsilon$	Differentialoperator (Definition 3.4)
$\mathbf{F}$	Kraftvektor (allgemein)
$\mathbf{F}_{VE}$	Resultierende Kraft eines Verbindungselementes
$\mathbf{F}_{GVE}$	Resultierende Kraft eines globalen Verbindungselementes
$\mathbf{F}_O$	Verformungskraft
$\mathbf{F}_G$	Schwerkraft
$\mathbf{G}$	G-Vektor (Definition 3.9)
$\mathbf{KP}$	Kontrollpunktmatrix
$\mathbf{M}$	Momentenvektor (allgemein)
$\mathbf{P}$	Kontrollpunktvektor
$\mathbf{T}_n$, $\mathbf{T}_m$	Transformationsmatrizen (auf globales Koordinatensystem)
$\mathbf{V}_T$	Koordinatensystem-Transformationsmatrix
$\mathbf{Z}$	Z-Vektor (Definition 3.10)
$\mathbf{n}$	Normalenvektor
$\mathbf{p}$	Knotenpositionsvektor
$\mathbf{r}$	Ortsvektor der translatorischen Freiheitsgrade eines Körpers/Punktes
$\mathbf{s}$	Stützstellenvektor
$\mathbf{u}$	Relativer Zustandsvektor (translatorisch)
$\mathbf{v}$	Geschwindigkeitsvektor
$\mathbf{x}$	Systemvektor eines Körpers/Punktes
φ	Ortsvektor der rotatorischen Freiheitsgrade eines Körpers/Punktes
ϕ	Relativer Zustandsvektor (rotatorisch)
ε	Verzerrungsvektor
σ	Spannungsvektor

Skalare Größen werden in Normalschrift dargestellt. Die Bezeichner von Vektorelementen entsprechen denen der Vektoren mit zusätzlichen Indizes (meist mit x, y, z bezeichnet):

A	Fläche eines Körpers (Oberfläche)
V	Volumen eines Körpers
e	Eingangsgröße für Verbindungselement-Transformation
pb_val	Plausibilitätswert der Interaktion
s, t	Parametervariablen von Splines
s_I	Interaktionsschranke
s_{NV}	Stufigkeit von Integrationsalgorithmen
t	Zeit (absolut)
Δe	Eingangsgröße für Verbindungselement-Grundfunktion
ε	Dehnung
γ	Schubverzerrung
σ	Normalspannung
τ	Tangentialspannung
λ	Eigenwert
σ_s	Stabilitätsfaktor von Integrationsverfahren

Funktionen werden stets kursiv dargestellt:

$C()$	Kurvenfunktion (parametrisch)
$S()$	Flächenfunktion (parametrisch)
$\boldsymbol{D}_K()$	Dämpfungsfunktions-Vektor
$\boldsymbol{F}()$	Knotenkraftfunktions-Vektor
$d()$	Dämpfungsfunktion
$f()$	Kraftfunktion
$f_G()$	Grundfunktion eines Verbindungselements
$f_{Hyp}()$	Wirkende Kraft des Hyperknoten
$f_{KK}()$	Übertragene Kraft an Kindknoten
$f_O()$	Verformungskraft

Parameter und **Konstanten** werden durch kursive Zeichen gekennzeichnet:

E	Elastizitätsmodul
$\boldsymbol{B}$	Beschränkungsmatrix (Knotenfreiheitsgrade)
$\boldsymbol{C}$	Stoffmatrix
$\boldsymbol{D}$	Dämpfungsmatrix
$\boldsymbol{M}$	Massenmatrix
$\boldsymbol{J}$	Hauptträgheitsmatrix
d	Dämpfungskonstante
g	Erdbeschleunigung
h	Schrittweite (Numerische Integration)
k	Ersatz-Federkonstante/Elementkonstante
m	Masse
par	Parameterliste, allgemein
ρ	Massendichte
ν	Poissonzahl
τ	Systemzeitkonstante

Abkürzungen

CAD	**C**omputer **A**ided **D**esign
DGL	**D**ifferential**gl**eichung
FE	**F**inites **E**lement
FEM	**F**inite-**E**lemente-**M**ethode
F-Knoten	'**F**reier' Masseknoten (Definition 3.11)
G-Knoten	Masseknoten mit **G**eometriebezug (Definition 3.9)
GVE	**G**lobales Verbindungs**e**lement
IS	**I**nteraktions**s**trecke
IW	**I**nteraktions**w**irkungsvektor
KISMET	**Ki**nematic **S**imulation, **M**onitoring and Off-Line Programming **E**nvironment for **T**elerobotics
KK	**K**ind**k**noten
KP	**K**ontroll**p**unkt eines Splines/Freiformfläche
KS	**K**oordinaten**s**ystem
MESD	Methodik der **M**odellbildung zur **e**chtzeitfähigen **S**imulation und Manipulation **d**eformierbarer Objekte über physikalisch-basierte nodale Systeme
MIC	**M**inimal-**I**nvasive **C**hirurgie
NK	**N**achbar**k**noten
NURBS	**N**on-**U**niform **R**ational **B**-Spline **S**urface
VE	**V**erbindungs**e**lement
VR	**V**irtuelle **R**ealität/'**V**irtual **R**eality'
VK	**V**ater**k**noten
Z-Knoten	'**Z**entral'-Masseknoten (Definition 3.10)

1 Einleitung

1.1 Motivation

Der Begriff 'Virtuelle Realität' (VR)[1] bezeichnet ein relativ neues Anwendungsgebiet der interaktiven Computergraphik. VR-basierte Systeme halten Einzug in Wissenschaft, Industrie, Unterhaltung, Kunst und auch in die Medizin [Voe95]. Eine genaue Begriffsbestimmung ist sehr schwierig und auch nicht Ziel dieser Arbeit. Trotzdem soll an dieser Stelle zur Erläuterung der Motivation ein kurzer Überblick erfolgen.

Mit 'Virtuelle Realität' wird eine besonders intuitive Art der Mensch-Maschine-Kommunikation beschrieben. Kennzeichnend hierfür sind:

- Die Erzeugung dreidimensionaler künstlicher Welten als Nachbildung der Realität mit Hilfe eines Rechnersystems.
- Die Einbindung des Menschen in diese virtuelle Welt mit visueller, teilweise auch taktiler und akustischer Stimulierung (**Immersion**: *Empfindung des Menschen, sich in einer realen Welt zu befinden*).
- Die Beeinflussung der künstlichen Welt durch direktes Eingreifen des Menschen (**Interaktion**: *aufeinanderbezogenes Handeln zwischen Mensch und virtuellen Objekten*).

Man kann in der VR verschiedene Stufen abgrenzen, die sich insbesondere in der Art der Interaktion und damit auch im Grad der Immersion unterscheiden:

Die erste Stufe ist charakterisiert durch ein festes, unveränderliches Szenario. Die Interaktion beschränkt sich auf das Verändern des Blickstandortes und der Blickrichtung, der Mensch kann sich im virtuellen Raum 'bewegen'. Ein Beispiel zeigt Bild 1.1, eine Momentaufnahme des Operationssaals der Zukunft[2] für die Minimal-Invasive Chirurgie (MIC)[3].

In einer weiteren Immersionsstufe ist das interaktive Verändern des virtuellen Szenarios möglich, beispielsweise das Greifen von Objekten und Ablegen an einer anderen Stelle sowie

[1] Im deutschen Sprachschatz ist oftmals auch der englische Originalausdruck 'Virtual Reality' gebräuchlich.
[2] Das Modell wurde unter Mitwirkung des Autors im Rahmen eines Projektes am Forschungszentrum Karlsruhe erstellt.
[3] Der Begriff und die Technik der 'Minimal-Invasiven Chirurgie' wird in Kapitel 6 genauer erläutert.

das Bewegen von virtuellen Systemen. Im Szenario von Bild 1.1 können die Bewegungen der MIC-Manipulatoren, Endoskope und OP-Lampen direkt gesteuert werden.

Eine weiter Steigerung der Immersion ist möglich, wenn die virtuellen Objekte oder Systeme selbst ein reales Verhalten zeigen, also eine wirklichkeitsgetreue **Simulation** (*Nachbildung oder Nachahmung technischer/naturwissenschaftlicher Vorgänge*)[4] dem VR-Szenario unterlagert ist. Die Interaktion wird damit besonders realistisch, da die Reaktionen der simulierten Objekte den Erwartungen des Menschen entsprechen.

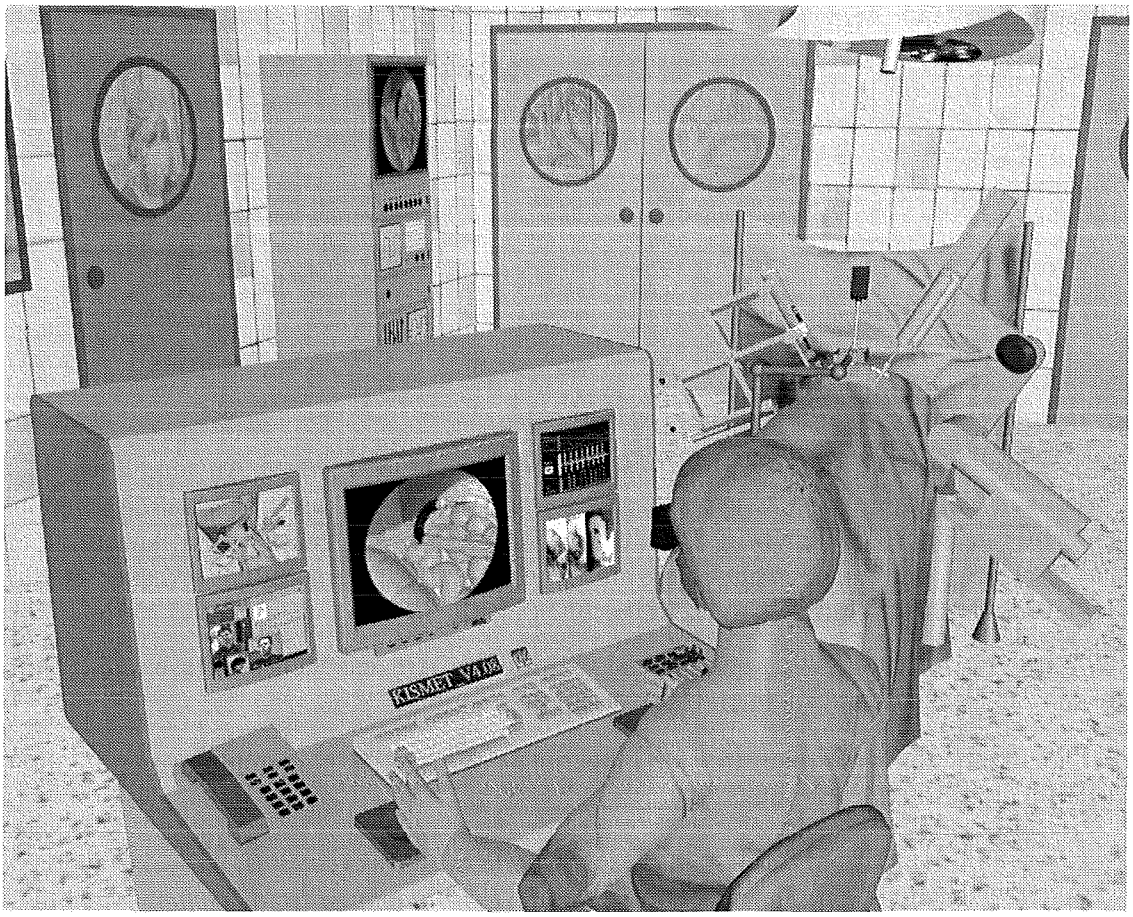


Bild 1.1: VR-Szenario - 'Operationsaal der Zukunft'

1.2 Problemstellung und Begriffsbestimmung

Die Simulation und Darstellung starrer Körper hat sich in der Computergraphik und anderen technischen Gebieten etabliert. Mit der heutigen Rechnergeneration ist eine Simulation der Bewegungen in Echtzeit auch bei komplexeren Modellen möglich, solange sich das Szenario auf feste, unveränderliche Geometrien beschränkt. Deformierbare Objekte hingegen sind morphologisch variabel, wodurch die Simulation der inneren und äußeren Bewegung (Kinematik) und der Dynamik ungleich schwieriger ist. Die klassische Finite-Elemente-Methode bietet hierzu Lösungen an, diese sind jedoch wegen der großen Anzahl an inneren Freiheitsgraden weit von der Echtzeitfähigkeit entfernt.

[4] Die *kursiv* gekennzeichneten Definitionen wurden zum großen Teil aus dem 'Duden' entnommen.

Die vorliegende Arbeit beschäftigt sich mit der Repräsentation deformierbarer Objekte innerhalb einer interaktiven graphischen VR-Simulationsumgebung. Das zentrale Problem besteht in einer möglichst realitätsgetreuen dreidimensionalen Bewegungs- und Verhaltenssimulation unter Einwirkung äußerer Anregungen. Dies beinhaltet - genau genommen - die Bestimmung der Bewegungsbahnen aller Objektpunkte.

Wichtige Anforderung für Interaktivität ist die Möglichkeit der direkten Einflußnahme auf das Simulationsszenario und die unmittelbare Reaktion der betroffenen Objekte. Bedingung hierzu ist die **Echtzeitfähigkeit**[5], worin auch die zentrale Forderung für die in dieser Arbeit entwickelten Verfahren liegt. Die Bildberechnung und Simulation muß *schritthaltend* mit den äußeren Anregungen und Abläufen arbeiten und entsprechend konforme Ergebnisse liefern. Insbesondere die Analogie zwischen zeitlichen Abläufen in der Realität und im Simulationsszenario muß gewährleistet sein.

Im Gegensatz zur VR spielt die Echtzeitfähigkeit bei der **Animation** (*Belebung von Objekten*) keine Rolle, beispielsweise im Trickfilm. Hier werden die Bildsequenzen einzeln berechnet, gespeichert und erst beim Animationsablauf verkettet und dargestellt. Es erfolgt also eine zeitliche Entkopplung zwischen Bildberechnung (auch mit Hilfe von Simulationen) und Bildausgabe. Die **Visualisierung** (*auf optisch ansprechende Weise darstellen*) beinhaltet die graphische Repräsentation der Simulationsergebnisse, hier in Form eines VR-Szenarios. Eine geeignete Ausgabeeinheit kann den Grad der Immersion wesentlich erhöhen.

Die Simulation benötigt zwingend die Erstellung eines **Modells** (*abgebildetes oder veranschaulichtes Objekt, durch Modellieren geschaffene Gestalt*) des realen Systems. Der Vorgang der spezifischen Modellerstellung wird als **Modellierung** bezeichnet, die übergeordnete Methodik als **Modellbildung**. Zur Simulation auf einem Rechnersystem wird ein abstraktes Modell benötigt, das die interessierenden Zusammenhänge mathematisch oder formal beschreibt.

Ein essentieller Begriff der Interaktionsbehandlung ist die **Kollision** (*Zusammenstoßen*) von Objekten im virtuellen Raum, die im allgemeinen zur Detektion eines Interaktionsvorganges herangezogen wird. Der Einfluß des Menschen auf ein ausgewähltes Objekt wird als **Manipulation** (*Handhabung*) bezeichnet. Hierbei wird auf die Objektdynamik eingewirkt und eine entsprechende Deformation verursacht. Eine weitergehende Wirkung wird durch die Modell-**Modifikation** (*Abänderung, Abwandlung*) realisiert, bei der die Objekte nicht nur einer morpholgischen Änderung durch Verformung, sondern auch einer strukturellen Modellwandlung unterzogen werden.

1.3 Zielsetzung und Anwendungen der Arbeit

Ziel der Arbeit war die Entwicklung eines echtzeitfähigen Verfahrens zur dynamischen Simulation deformierbarer Objekte. Hierbei sollen neben den geometrischen Formen auch die physikalischen Eigenschaften gegebener Körper möglichst gut nachgebildet werden. Die Echtzeitfähigkeit ist hierbei die wichtigste Bedingung, da die Objekte interaktiv manipulierbar sein sollen.

Mit dem MESD-Verfahren *(Methodik der Modellbildung zur echtzeitfähigen Simulation und Manipulation deformierbarer Objekte über physikalisch-basierte nodale Systeme)* werden vorhandene Ansätze aus der Animationstechnik und Mechanik integriert und weiterentwickelt, um die mechanischen Eigenschaften deformierbarer Objekte möglichst gut zu approximieren und eine Simulation und Manipulation in Echtzeit zu ermöglichen. Die

[5] Ein häufiges Synonym zu 'Echtzeit' ist die 'Realzeit' oder auch das englische 'Real-Time'.

Simulation der physikalischen Eigenschaften erfordert die Erstellung eines äquivalenten mathematischen Modells, wobei hauptsächlich von Sachverhalten der Mechanik ausgegangen wird.

Zentraler Bestandteil der Interaktion ist das Eingreifen des Bedieners und die Manipulation der Modelle. Hierzu sind Verfahren zur echtzeitfähigen Kollisionserkennung sowie zur Auswertung der Interaktion in Abhängigkeit der kollidierenden Körper zu entwickeln. Insbesondere die Reaktion der deformierbaren Objekte auf die Manipulation ist zu spezifizieren und zu simulieren. Bei Objektmodifikationen ist eine Strukturänderung der MESD-Modelle auszuführen, wichtig hierbei ist eine effiziente Datenstruktur. Neben den physikalischen Eigenschaften werden auch Verhaltensmodelle benötigt.

Die graphische Repräsentation besteht in der Visualisierung der Simulationsergebnisse auf einem Bildschirm in Form von geometrischen Modellen der deformierbaren Objekte. Ziel ist ein möglichst realistisches Aussehen. Hier werden zwei Ansätze vorgestellt, ein Ansatz auf Basis von Freiformflächen sowie eine Darstellung durch polygonale Netze.

Anwendungen

Die erarbeiteten Verfahren sind Grundlage für die Entwicklung des 'Karlsruher Endoskopie-Trainers', mit dem die Ausbildung und das Training in der Minimal-Invasiven Chirurgie (MIC) sinnvoll unterstützt werden kann. Das Operationsfeld wird hierbei durch Modelle auf Basis deformierbarer Objekte nachgebildet, wobei der Bediener über eine realistische Bedienerschnittstelle interaktiv Einfluß auf das Szenario nehmen kann. Typische Operationsvorgänge können so in Echtzeit simuliert werden. Die Sicht der Endoskopkamera wird vom Simulationssystem virtuell erzeugt.

Ein prototypischer Demonstrator wurde für den Bereich der Cholezystektomie (Gallenblasenentfernung) aufgebaut. Über eine Phantombox als realitätsnahe Eingabeschnittstelle sind typische chirurgische Tätigkeiten wie Greifen, Klammern, Schneiden und Koagulieren an den Organmodellen möglich.

Weitere Anwendungsgebiete liegen in den Feldern der interaktiven Computergraphik und Simulationstechnik, in denen die Echtzeitbedingung eine besondere Rolle spielt, oder auch in der Animationstechnik. Beispiele sind 'Virtual Reality'-Anwendungen, die Simulation deformierbarer technischer Systeme und die schnelle Rekonstruktion von Objekten aus Meßdaten.

1.4 Übersicht über die folgenden Kapitel

In Kapitel 2 wird das Problem und die daraus resultierende Aufgabenstellung spezifiziert. Es wird der Stand der Technik vorgestellt und ein Überblick über die betreffende Literatur gegeben. Weiterhin wird das als Implementierungsbasis dienende Simulationssystem 'KISMET' kurz eingeführt.

Der Hauptteil dieser Arbeit gliedert sich in drei Bereiche. Die physikalische Repräsentation deformierbarer Objekte wird in Kapitel 3 abgehandelt, hierbei wird die Modellbildung und Simulation mit dem MESD-Verfahren erarbeitet. In Kapitel 4 werden die Stufen der Interaktionen und Verfahren zur Objektmanipulation vorgestellt. Die graphische Repräsentation, die auf den Methoden der geometrischen Modellbildung basiert, wird in Kapitel 5 behandelt.

Als Beispiel für den Einsatz der entwickelten Verfahren wird in Kapitel 6 der 'Karlsruher Endoskopie-Trainer' ausführlich vorgestellt. Weitere Anwendungsfelder werden in Kapitel 7 angesprochen, hier erfolgt auch die Diskussion der Ergebnisse sowie ein Ausblick auf weiteres Entwicklungspotential.

2 Problemspezifikation und Stand der Technik

Die Thematik dieser Arbeit besitzt interdisziplinären Charakter, sie zeigt viele Überschneidungen mit Methoden aus der Informatik, der Ingenieurwissenschaften und der Mathematik. In diesem Kapitel soll eine genaue Analyse der Problemstellung erfolgen, hierbei werden Querverweise zu verwandten Verfahren gezogen und bekannte Methoden vorgestellt.

2.1 Analyse der Problemstellung

Kernziel der vorliegenden Arbeit ist die möglichst realistische Repräsentation deformierbarer Objekte. Diese sind charakterisiert durch sowohl eine innere, lokale Dynamik (Elastodynamik) des Körpers selbst als auch durch eine äußere, globale Bewegung (Kinematik) gegenüber der Umgebung.

Bemerkung 2.1

Die **Repräsentation** im Sinne dieser Arbeit umfaßt sowohl die *Modellbildung*, die *Simulation* (Kinematik, Elastodynamik und Verhalten bei Interaktionen) als auch die *Visualisierung* deformierbarer Objekte

Ein wichtiger Aspekt ist hierbei die interaktive Einflußnahme auf die Simulation. Nach intuitivem Empfinden ist eine direkte Interaktion möglich, wenn die systembezogene Verzögerung von Stimulation (Handlung des Menschen) bis zur Reaktion (Wahrnehmung durch den Menschen) höchstens 0,2 Sekunden beträgt. Weiterhin ist für das realistische Empfinden der dynamischen Vorgänge eine Bildwiederholrate von mindestens 10-15 Bildern in der Sekunde notwendig[1]. Diese Werte sind nur Abschätzungen, sie differieren nach subjektivem Empfinden von Mensch zu Mensch.

Für eine echtzeitfähige Simulationsumgebung bedeutet dies, daß die gesamte Synthese eines neuen Bildes innerhalb einer maximalen Zeitspanne von 0,1 Sekunden ausgeführt werden muß. Dies beinhaltet die Umsetzung der stimulierenden Daten, die Auflösung der kinematischen Strukturen, Berechnung der Elastodynamik, Interaktionsbehandlung und die eigentliche graphische Datenverarbeitung bis hin zur Graphikausgabe. Die entwickelten

[1] Die Bildwiederholrate ist die Anzahl der von der Simulation neu generierten Bilder pro Sekunde. Diese Rate ist zu unterscheiden von der Auffrischrate des Monitorbildes, die stets bei 60-120 Hz liegt und unabhängig von der eigentlichen Simulation ist.

Verfahren zur Repräsentation deformierbarer Objekte waren also hinsichtlich Rechenzeit und Effizienz zu optimieren.

Bemerkung 2.2

Die entscheidende Anforderung an die erarbeiteten Verfahren ist die **Echtzeitbedingung**, da ansonsten keine Interaktivität gewährleistet werden kann.

Die Echtzeitanforderung steht dem Ziel nach möglichst hoher Realitätsnähe entgegen. In dem Spannungsfeld der verschiedenen, teilweise konträren Anforderungen sind unter Beachtung des technisch machbaren stets Kompromisse zu schließen, die je nach Anwendungsfall verschieden gewichtet sein können. Deutlich wird diese Problematik anhand des Anforderungsdreiecks (Bild 2.1). Neben der Effizienz der Verfahren ist für die Erfüllung der Ansprüche natürlich auch die verwendete Rechnerhardware von entscheidender Bedeutung.

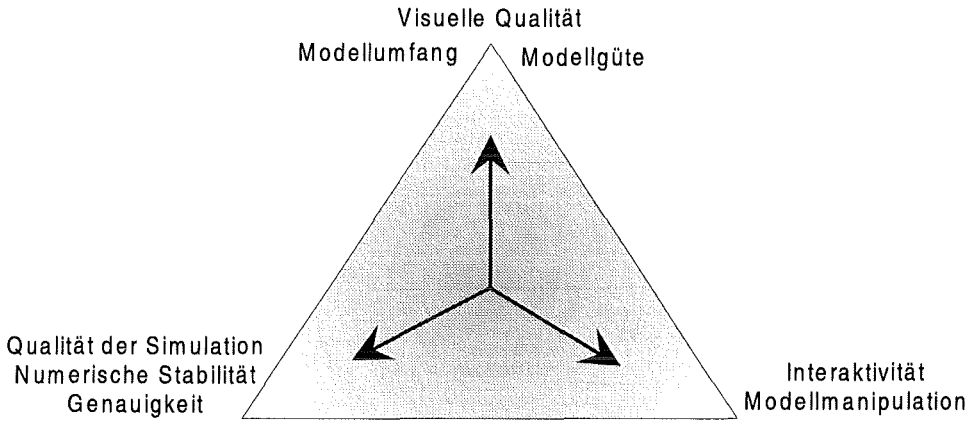


Bild 2.1: Spannungsdreieck der gegensätzlichen Anforderungen

Analysiert man die Thematik genauer, so kann man die gegebene Aufgabenstellung in drei Teilgebiete untergliedern, die verschiedene partielle Lösungsbereiche abdecken. Man kann diese Bereiche getrennt voneinander betrachten und herausarbeiten, für eine effiziente Simulation ist jedoch die Integration zu einem homogenen System unabdingbar (Bild 2.2). Deshalb müssen die Schnittstellen genau spezifiziert und aufeinander abgestimmt werden.

- Die **'Graphische Repräsentation'** umfaßt die geometrische Modellbildung des Körpers, die Spezifizierung der graphischen Darstellung bis zur Ausgabe auf einem Bildschirm. Hierzu gehören auch die Definition der Oberflächeneigenschaften (Texturen), die Lichtmodelle und Schattierungsverfahren. Die graphische Repräsentation wird in Kapitel 4 erläutert.

- Die **'Physikalische Repräsentation'** beinhaltet die Nachbildung der physikalischen Eigenschaften des Körpers. In dieser Arbeit werden die physikalischen Eigenschaften hauptsächlich auf das mechanische Verhalten beschränkt, insbesondere die Dynamik und Kinematik der deformierbaren Objekte. Von großer Bedeutung ist hierbei die elastodynamische Modellbildung. Das hierfür entwickelte MESD-Verfahren wird in Kapitel 3 ausführlich vorgestellt.

- Mit **'Interaktion'** wird die Beeinflussung des Modellszenarios und der Objekte zur Simulationslaufzeit beschrieben. Hierzu gehört neben der Detektion einer Interaktion (Kollisionserkennung) auch die betreffende Auswertung. Diese beinhaltet die entsprech-

ende Objektmanipulation sowie eventuell eine Modifikation der graphischen und physikalischen Modelle. Dieser Problemteil ist in Kapitel 4 näher erläutert.

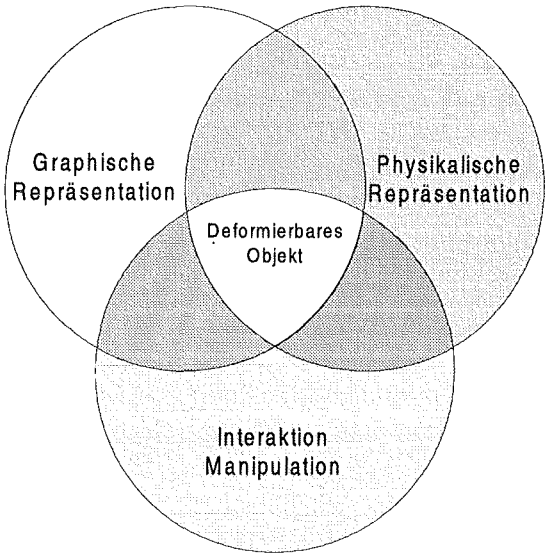


Bild 2.2: Struktur der Repräsentation deformierbarer Objekte

2.2 Stand der Technik - Diskussion der Anforderungen

Ein homogener Lösungsansatz der ganzheitlichen Problemstellung wurde bisher - zumindest nach Wissen des Autors - noch nicht versucht. Vielmehr beziehen sich bekannte Verfahren auf einzelne Teilgebiete dieser Arbeit, die hier auch einzeln betrachtet und diskutiert werden sollen. Als gute Untergliederung bietet sich die Struktur von Bild 2.2 an, die zwischen graphischer und physikalischer Repräsentation sowie Interaktion unterscheidet. Einen Überblick über Methoden auf diesen drei Gebieten wird in den folgenden Unterkapiteln gegeben.

2.2.1 Graphische Repräsentation

In der graphischen Datenverarbeitung, insbesondere in der CAD[2]-Technik, sind verschiedene Methoden der räumlichen Modellierung und Repräsentation geometrischer Objekte gebräuchlich [FDF90, HL89]. Im wesentlichen lassen sich die Verfahren in drei Gruppen einteilen:

- **CSG-Modellierung** (Constructive Solid Geometry)

  Ein räumliches Objekt wird durch eine Anzahl von geometrischen Grundelementen (sogenannte 'Primitive', beispielsweise Quader, Zylinder, Kugel, Kegel) und deren logische Verknüpfung (Vereinigung, Differenz, Schnittmenge) beschrieben [RV83].

- **B-REP-Modellierung** (Boundary Representation)

  Die Repräsentation des Objektes beruht auf der Darstellung der Oberflächen des räumlichen Körpers, also der Flächen, die die Geometrie nach außen hin begrenzen. Diese Modellierung läßt sich wiederum in zwei Typen untergliedern: einer *exakten* und einer *angenäherten* Darstellung. Bei dem exakten Typ wird das Objekt durch analytische

[2] Computer Aided Design - Rechnerunterstützte Planung, Gestaltung und Konstruktion

Gleichungen, also durch mathematisch genau definierte stetige Flächen beschrieben. Beispiel hierfür sind Freiformflächenmodelle auf Basis von Splines [Far88, BBB87]. Bei dem angenäherten Typ wird das Objekt hingegen durch Kanten ('Drahtgittermodell') oder ebene Flächenstücke dargestellt [SGI91]. Die Gesamtheit der Kanten oder Flächenstücke formt die Objekt-Geometrie (Polyeder-Modell).

- **Volumen-Modelle**

  Diese Repräsentation stützt sich auf die Unterteilung räumlicher Körper in diskrete Teilelemente und deren zusammenhängende Darstellung. Beispiele hierzu sind Verfahren zur direkten Visualisierung von sogenannten Voxeln, den kleinsten Volumenelementen eines Körpers [DCH88]. Zu dieser Gruppe gehört auch die Octree-Methode [JT80], die zur effizienteren Darstellung eine hierarchische Baumstruktur der Volumenelemente verwendet.

Diese Geometrie-Repräsentationen sind in einer höheren Modellierungsebene angeordnet. In üblichen Graphikrechnern findet auf unterer Ebene (Graphikhardware oder -software) stets eine 'on-line' Umwandlung der CSG-Modelle und exakten B-REP-Darstellung in eine approximierte Darstellung durch ebene Flächenstücke statt. Die eigentliche Graphikausgabe erfolgt dann meist in Form von polygonalen Netzen. Die Art und Genauigkeit dieser rechnerinternen Geometrie-Transformation kann vorgegeben werden.

Tabelle 2.1: Kriterien und Anforderungen der geometrischen Modellbildung

	Kriterien und Aspekte zum Einsatz der geometrischen Modellbildung	***Anforderungen gemäß der Aufgabenstellung***
1	**Rechenzeit** für die Datenaufbereitung bis zur Graphikausgabe	Wichtigstes Kriterium: möglichst geringe Verarbeitungszeit, da Echtzeitanforderung
2	**Qualität** der Darstellung (visuelle Qualität, Approximation des realen Objekts)	Möglichst gut und realistisch, hat aber keine Priorität Wichtiger: Anpassung der Genauigkeit, Möglichkeit der lokalen Qualitätsbeeinflussung/Glättung
3	**Darstellbarkeit** von Objekten (gute Repräsentation der Art der geometrischen Form: Kanten/Rundungen)	Deformierbare Objekte sind im allgemeinen durch stetige Oberflächen (C^0- und C^1-stetig) charakterisiert ('glatt'). Dieser Typ soll besonders gut repräsentiert werden
4	**Formänderungen** durch Beeinflussung der Geometrieparameter	Formänderung sollen keine völlige Neudefinition der Geometrie zur Folge haben, lokale Änderungen dürfen nur lokale Auswirkungen besitzen
5	**Modifikation** der Geometrien, Strukturänderungen und Erweiterungen	Wichtig für Modellmodifikationen: die rechnerinterne Repräsentation (Datenstruktur) soll schnell und einfach bearbeitbar sein, um interaktive Manipulationen zu gewährleisten, keine Neuberechnung der gesamten Geometrie auf hoher Ebene
6	**Speicherbedarf** der Datenstrukturen	Bei den heutigen Rechnerarchitekturen weniger relevant (außer bei Voxel-Modellen)

Die wichtigsten Aspekte und Kriterien der geometrischen Modellbildung sind den Anforderungen bezüglich der Problemstellung dieser Arbeit in Tabelle 2.1 gegenübergestellt. Eine grobe Bewertung der verschiedenen, oben kurz angesprochenen Modellierungsarten wird in Tabelle 2.2 vorgenommen. Das Kriterium 'Qualität' ist im wesentlichen direkt von der Modellierungsgüte abhängig, wichtig bei der Wertung war hier die Möglichkeit der Qualitätsbeeinflussung durch Verfeinerung, Vergröberung oder Glättung.

In der interaktiven Computergraphik und VR-Anwendungen werden hauptsächlich approximierte Oberflächenmodelle (polygonale Netze) verwendet, die jedoch häufig über andere Modellierungsmethoden erzeugt und anschließend 'off-line' umgesetzt werden. In der CAD-Technik und in Animationen kommen neben den Polyedermodellen zunehmend auch

Freiformflächen zum Einsatz. Volumenmodelle haben ihre Anwendungsgebiete meist in der reinen Visualisierung von Volumendaten.

Tabelle 2.2: Eignung der geometrischen Modellbildungsarten bezüglich der Anforderungen

	CSG	Freiformfläche	Polyeder	Voxel	Octree
1: Rechenzeit	-	-	++	--	-
2: Qualität	+	++	+	o	o
3: Darstellbarkeit	--	+	+	+	+
4: Formänderung	--	++	+	o	o
5: Modifikation	--	-	+	o	+
6: Speicherbedarf	++	+	o	--	-

++ : sehr gut geeignet, + : geeignet, o : bedingt geeignet, - : mäßig geeignet, -- : ungeeignet

Die graphische Repräsentation besteht nicht nur aus der geometrischen Modellbildung, sondern auch aus den definierten optischen Eigenschaften der dargestellten Körperoberflächen. Der Einsatz der im folgenden beschriebenen Methoden ist jedoch bei Echtzeitanwendungen nur praktikabel, wenn die Graphikhardware dies effizient unterstützt.

- **Darstellungs- und Schattierungsmodelle** - diese beinhalten die Zuweisung von Materialeigenschaften wie Grundfarbe und Oberflächenattribute (diffuse und spiegelnde Reflektion, Transparenz) sowie die Art der Schattierungsberechnung [FvD84]. Das 'Flat-Shading'-Verfahren berechnet eine konstante Farbintensität für jede darzustellende Facette, bei gekrümmten Flächen treten an den Facettenübergängen deshalb Unstetigkeiten auf. Bei dem 'Gouraud-Shading' [Gou71] werden die Farbintensitäten an allen Polygoneckpunkten der jeweiligen Facetten berechnet, die dazwischenliegenden Farbintensitäten werden interpoliert. Die Stetigkeit der Farbübergänge an den Facettenkanten wird hierbei gesichert. Das 'Phong-Shading' [BT75] ist noch rechenaufwendiger, hierbei werden die zur Berechnung der Farbintensitäten notwendigen Normalenvektoren zwischen den Polygoneckpunkten interpoliert. Bei stark reflektierenden Oberflächen ergibt sich hierbei eine weitere qualitative Verbesserung der Schattierungsdarstellung. Für eine fotorealistische Darstellung von virtuellen Szenarien sind aufwendigere Verfahren wie Ray-Tracing [Whi80] oder Ray-Casting [Rot82] notwendig. Diese Methoden sind jedoch weit vom Echtzeitanspruch entfernt und deshalb für eine interaktive Simulation nicht geeignet. Radiosity-Verfahren [KSH95, SM95] versprechen schnellere Berechnungen, allerdings nur bei unbewegten Objekten oder Lichtquellen.
- **Beleuchtungsmodelle** - Einfügen und Spezifikation von Lichtquellen in das Simulations-Szenario, Definition von diffusem Licht, Punkt- und Parallellichtquellen oder Spots [FDF90].
- **Texturierung** - Projektion von Bitmaps[3] auf die Oberfläche des geometrischen Körpers [Hec86, MYV93]. Hierbei muß jedem Oberflächenpunkt ein Feldelement des Bitmaps zugeordnet werden. Dies geschieht meist durch direkte Zuweisung charakteristischer Oberflächenpunkte und automatische Interpolation der Zwischenwerte.

2.2.2 Physikalische Repräsentation

Die physikalische Repräsentation beinhaltet die Nachbildung des physikalischen Verhaltens deformierbarer Objekte, hierbei insbesondere der mechanischen Eigenschaften. Wesentlicher Schritt hierzu ist die Aufstellung eines physikalischen/mathematischen Modells des Objektes.

[3] Zweidimensionales Feld mit Farbwerten/Grauwerten ('Bild').

- Der bekannteste Ansatz hierzu ist die **Finite-Elemente-Methode** (FEM), die sich in Mechanik und Maschinenbau fest etabliert hat [Standardwerke: BW76, ZM83, KW91]. Das Verfahren beruht auf einer diskreten Modellbildung, also der Diskretisierung eines Körpers in mehrdimensionale Basiselemente ('Finite Elemente'). Innerhalb dieser Elemente kann das mechanische Verhalten durch spezifische funktionale Ansätze, den Formfunktionen, beschrieben werden. Aus dem Zusammenfügen der verschiedenen finiten Elemente ergeben sich die Übergangs- und Randbedingungen. Mathematisch bedeutet dies, die problembeschreibenden, komplexen und partiellen Differentialgleichungen in ein System von einfachen, linearen Differentialgleichungen umzuwandeln[4]. Aufgrund der großen Anzahl an inneren Freiheitsgraden und Modellparametern wird das resultierende Gleichungssystem im allgemeinen sehr groß, so daß die FE-Methode weit vom Echtzeitanspruch entfernt ist. Eine Einführung zur Verwendung von FE-Verfahren bei Animationen gibt Pentland [PS90]. Anwendungen bezüglich biomechanischer Objekte finden sich bei Larrabee [Lar86], der Hautverformungen berechnet, sowie bei Chen und Zeltzer [CZ92], die Muskeleinwirkungen auf Körperteile untersuchen. Gourret [GTT89] bestimmt Hautverformungen bei Einfluß äußerer Kräfte.

- Eine Vereinfachung und damit eine schnellere Berechnung ermöglicht das **Finite-Differenzen-Verfahren (FD)** [Grundlagen: Sch88, BP91, BS87]. Dieses Verfahren beruht auf der Auswertung der problembeschreibenden Differentialgleichungen an äquidistanten Stellen, also einer Zerlegung in homogene Teilräume. Hieraus ergeben sich für jeden Teilraum entsprechende Differenzengleichungen. Nachteil ist neben der immer noch hohen Rechenzeit auch die starre und schlecht anzupassende Modellbildung. Die FD-Methode wird von Terzopoulos [TPB87, TW88] zur Animation deformierbarer Körper eingesetzt.

- **Nodale Netzmodelle** können als weitere Vereinfachung der FEM angesehen werden. Hierbei wird die Masse des Körpers in nulldimensionale, massebehaftete Knoten diskretisiert. Jeweils ein Paar dieser dynamischen Masseknoten wird über kraftübertragende Verbindungselemente verknüpft, meist lineare Federelemente. Das resultierende Differentialgleichungssystem ist relativ einfach aufzustellen und zu lösen, so daß dieser Ansatz dem Echtzeitanspruch am nächsten kommt. Eine Übersicht zur Thematik wurde von Deussen aufgestellt [Deu96]. Nodale Netze wurden von Terzopoulos zur Animation von Gesichtern und Gesichtsbewegungen (Mimik) eingesetzt [TW90, TW91], Waters verwandte die Methodik auch zur Simulation von Gesichtsmuskeln [Wat92].

Aus der Animationstechnik wurden auch weitere interessante Verfahren bekannt. Ziel war es hier, bei aufeinanderfolgenden Bildsequenzen die Modelländerungen nicht einzeln auf unterster Ebene durchführen zu müssen, sondern eine Deformation auf höherer Ebene zu definieren. Dies verspricht neben einem realistischeren Verhalten auch eine Zeitersparnis bei der Animationserstellung. Kommerzielle Animationssysteme wie '*SOFTIMAGE* 3D' oder '*Alias* | *Wavefront* Studio' setzen solche Verfahren ein.

- **Partikelsysteme** bestehen aus punktförmigen Objekten ('Partikeln'), die in funktionalem Bezug zueinander stehen. Jeder Partikel hat Einfluß auf alle anderen Partikel des Systems und bestimmt somit deren Bewegung mit. Die Art des Einflusses wird durch Energiefunktionen und Regelsätze festgelegt [SZ92]. Zusätzlich kann eine Orientierung der Partikel festgelegt werden. Durch Agglomeration der Partikel im Raum ergeben sich die Oberflächen der deformierbaren Körper. Ein Beispiel ist die Animation von Wasser und anderen Flüssigkeiten, die Miller und Pearce [MP89] mit Partikelsystemen durchführen.

[4] Dynamischer Fall, im statischen Fall vereinfacht sich das Differentialgleichungssystem zu einem algebraischen Gleichungssystem

Reeves [Ree83] beschreibt die Anwendung von Partikelsystemen im Bereich der Filmanimation, womit Feuer und Explosionen effektiv simuliert werden können.

- **Deformationsfunktionen** verwenden zur Charakterisierung einer Verformung lokale Funktionen. Sehr gebräuchlich ist eine Modellierung auf Basis von Splines, so wie bei den Freiformdeformationen [SP86, HHY92]. Die Parameter der Funktionen bestimmen die Form der Geometrie, hierzu muß eine Beziehung der modellierten Körperelemente zu den Funktionswerten aufgestellt werden (Definition der Abhängigkeit). Barr [Bar84] verwendet hierfür sogenannte Transformationsfunktionen, der Ansatz von Lamousin [LW94] basiert auf NURBS-Techniken. Terzopoulos kombiniert die Verfahren mit Modellierungstechniken auf Basis von Superquadriken [TM91]. Für Zwecke der Graphik-Animation ist dieser Ansatz recht gut geeignet, für eine physikalische Simulation jedoch unpraktikabel.

Die hier vorgestellten Verfahren besitzen aufgrund des Einsatzes in den verschiedensten Anwendungsfeldern auch sehr konträre Eigenschaften. Um eine Bewertung vornehmen zu können, sollen zunächst die wichtigsten Anforderungen an die physikalische Repräsentation bezüglich der Aufgabenstellung herausgestellt werden:

Echtzeitfähigkeit: Die zur numerischen Berechnung der Dynamik notwendige Zeit muß kleiner als die dynamischen Zeitkonstanten der Simulation sein, um ein realistisches Verhalten zu erzielen. Gleichzeitig muß die strukturelle und numerische Stabilität stets gewährleistet bleiben, auch bei größeren Systemanregungen.

Qualität und Genauigkeit: Die gute Nachbildung der realen physikalischen Eigenschaften ist eine Hauptanforderung der Aufgabenstellung. Die Genauigkeit und das quantitativ korrekte Verhalten spielt hierbei jedoch eine untergeordnete Rolle, die Simulation soll qualitativ plausible Ergebnisse liefern.

Modellierungsfreiheit: Zur Modellierung deformierbarer Körper soll es möglichst wenig Einschränkungen geben. Es muß neben der reinen Oberflächenmodellierung auch eine Berücksichtigung der räumlichen Eigenschaften sowie eine lokale Anpassung der Modellgenauigkeit ermöglicht werden. Wichtig ist somit eine universelle Modellierbarkeit der realen mechanischen Gegebenheiten, auch mit Berücksichtigung von Anisotropien und lokalen Inhomogenitäten. Die physikalische Modellbildung sollte möglichst unabhängig von der graphischen Modellbildung und weiteren Anforderungen der Graphik sein.

Simulationsfreiheit: Die Implementierung verschiedenster mechanischer Charakteristiken soll möglich sein, also neben linearem elastodynamischen Verhalten auch nichtlineare Eigenschaften wie beispielsweise die Plastizität einbeziehen. Weiterhin soll keine Beschränkung der Freiheitsgrade erfolgen, insbesondere soll eine globale Bewegung der Körper innerhalb des Szenarios und eine dynamische Verknüpfung verschiedener Objekte ermöglicht werden.

Modifizierbarkeit: Die deformierbaren Objekte sollen zur Simulationslaufzeit in Struktur und Verhalten modifiziert werden können. Hierzu darf die Modifikation jedoch keine vollständige Neudefinition des Objektes erforderlich machen, sondern lediglich eine lokale Umorganisation. Wichtig hierfür ist eine effiziente Datenstruktur. Außerdem soll eine interaktive Anpassung der Parameter möglich sein.

Interaktionsmöglichkeit: Eine effektive Kollisionsprüfung sowie Modellmanipulationen müssen einfach und ohne großen Datenverarbeitungsaufwand durchführbar sein. Die Beaufschlagung lokaler Modellteile mit einer äußeren Kraft sowie eine Positionsbindung soll möglich sein.

Eine beurteilende Aussage bezüglich der Eignung der vorgestellten Methoden erfolgt in Tabelle 2.3. Kriterium ist hierbei nicht die Erfüllung der Anforderungen beim Stand der Technik, sondern die prinzipielle Eignung bei Weiterentwicklung der entsprechenden

Verfahren. Das Kriterium 'Qualität und Genauigkeit' ist wiederum von der Modellierungsgüte abhängig, wichtig bei der Wertung war hier die Möglichkeit der Qualitätssteigerung, Approximation der realen Eigenschaften sowie die Validierungsmöglichkeiten.

Tabelle 2.3: Eignung der physikalischen Modellbildungsarten bezüglich der Anforderungen

	Klassische FEM	Finite Differenzen	Nodale Netze	Partikel-systeme	Deformations funktionen
Echtzeitfähigkeit	--	o	+	o	-
Qualität und Genauigkeit	++	-	-	-	+
Modellierungsfreiheit	+	o	+	-	-
Simulationsfreiheit	+	+	+	-	--
Modifizierbarkeit	-	o	+	++	-
Interaktionsmöglichkeit	-	-	+	+	o

++ : sehr gut geeignet, + : geeignet, o : bedingt geeignet, - : mäßig geeignet, -- : ungeeignet

2.2.3 Interaktion, Kollisionsdetektion

Der erste Schritt der Interaktionsbehandlung ist die Kollisionserkennung - die Überprüfung, ob zwei oder mehr Körper einen gemeinsamen Teilraum im dreidimensionalen Modellszenario belegen. Hierfür gibt es insbesondere aus dem Bereich der Robotik schon Lösungen, die sich grob in vier Gruppen einteilen lassen. Eine gute Übersicht findet sich in [Gra87].

- **Analytische Kollisionsdetektionen**: sind die Körper komplett oder stückweise durch geschlossene Funktionen beschreibbar, so können eventuelle Schnitte (Punkte, Kurven, Flächen oder Volumen) analytisch berechnet werden [MW88, SWF93]. Bei komplexeren Geometrien werden die Gleichungssysteme jedoch kompliziert, die Lösung wird sehr aufwendig und nur numerisch möglich. Kollisionen einfacher Geometrien wie ebene Flächen ('Boden') oder Quader sind mit dieser Methodik leicht zu erfassen. Ein weiteres Beispiel ist das von Platt und Barr behandelte 'Constraints'-Verfahren [PB88].

- **Kollisionserkennung über Hüllvolumen**: nutzt die Vereinfachung der zu überprüfenden Geometrien durch abstrahierende Volumenelemente. Möglichst kleine geometrische Grundformen (meist Quader, Kugeln oder Zylinder), die die Körper ganz umschließen, werden vor der Simulationslaufzeit bestimmt. Mit diesen Hüllvolumen lassen sich vereinfachte, aber hierdurch schnelle Überschneidungstests durchführen [Dai88]. Genauer wird das Verfahren, wenn ein Objekt durch mehrere, verschachtelte Hüllvolumen repräsentiert wird [GAS94, Die89]. Ein Satz von Hüllvolumen beschreibt somit die zu untersuchenden Geometrien. Durch hierarchische Detaillierungsstufen wird das Verfahren sehr effizient.

- **Doppelbelegung von Volumenelementen**: dieses Verfahren ist besonders bei Körpern geeignet, die schon in Form von Volumenmodellen vorliegen. Falls ein Volumenelement ganz oder teilweise von zwei Objekten belegt wird, wird eine Kollision detektiert [UOT83].

- **Geometrische Testverfahren der Oberflächen**: finden insbesondere bei B-REP-Modellen Verwendung. Hierbei werden die polygonalen Flächenstücke, die beide Körper approximieren, auf gegenseitige Durchdringung geprüft. Die geometrische Auswertung ist prinzipiell zwar recht einfach, aufgrund der Vielzahl der zu überprüfenden Facetten und Schnittmöglichkeiten jedoch zeitlich sehr aufwendig [VT95].

Die Methoden lassen sich auch kombinieren, um die Effizienz zu steigern. Oft wird die Kollisionserkennung stufenweise durchgeführt, beispielsweise zuerst Hüllvolumen getestet

und bei positivem Ergebnis eine genauere Auswertung über geometrische Verfahren vorgenommen [Kue91].

Die wichtigste Anforderung an die Kollisionserkennung im Sinne der Aufgabenstellung ist wiederum die Echtzeitfähigkeit. Diese ist bei den aufwendigen Testverfahren der polygonalen Prüfung sowie der analytischen Detektion bei komplexeren Geometrien nicht gewährleistet. Zur effizienteren Kollisionserkennung muß oftmals eine Datenaufbereitung vor der Simulationslaufzeit durchgeführt werden, beispielsweise Hüllvolumen definiert und Datenfelder und -bäume generiert werden. Bei deformierbaren Objekten ist eine solche Vorverarbeitung jedoch nicht möglich, da die Körperform variant ist. Somit sind Volumenelementprüfungen und Hüllvolumentests nicht praktikabel, wobei letztere Methode allein auch zu ungenaue Ergebnisse liefert.

Ergebnis der Kollisionserkennung ist meist nur die Aussage, ob eine Überschneidung zweier Körper vorliegt oder nicht. Selten werden bei den Verfahren implizit die Berührungspositionen auf der Oberfläche bestimmt. Für Bahnplanungsaufgaben in der Robotik reicht dies aus, die Aufgabenstellung verlangt jedoch auch die genaue Bestimmung der interagierenden Regionen des Objekts sowie eine Auswertung und Reaktion auf die erkannte Kollision. Ansätze, die die resultierenden Kräfte berechnen und die Dynamik der kollidierenden Körper beeinflussen, gibt es beispielsweise von Moore und Wilhelms [MW88] oder von Hahn [Hah88], jedoch nur für die Starrkörperdynamik.

Im Bereich der Kollisionsauswertung bezüglich deformierbarer Objekte und Modellmanipulationen wurde bisher sehr wenig veröffentlicht. Einfache Ansätze finden sich bei der Simulation von Schnitten in Volumendaten [RRP96]. Die Modellmanipulation muß in Abhängigkeit der Art der Interaktion durchgeführt werden. Bestimmende Faktoren sind hier beispielsweise die Charakteristika der interagierenden Objekte, die ausgeübte Kraft und Eindringung, die betroffenen Objektteile sowie die Vorgeschichte der Interaktion. Relevant ist also insbesondere die Aktion und Einflußnahme des Benutzers auf das Simulationsszenario. Da nicht alle Arten von Interaktionen durch eine auf physikalischen Gesetzen basierende Algorithmik erfaßbar sind, ist die Einbeziehung von Verhaltensmodellen oder einer regelbasierten Methodik der Interaktionsauswertung notwendig.

2.3 Zusammenfassende Bewertung

Im letzten Abschnitt wurden bereits existierende Verfahren, die Teilaspekte der vorliegenden Aufgabengebiete behandeln, vorgestellt und hinsichtlich der Anforderungen der Problemstellung diskutiert. Hierbei wurden bezüglich der Relevanz und Verwertbarkeit für diese Arbeit unterschiedliche Vorarbeiten geleistet.

Auf dem Gebiet der *graphischen Repräsentation* sind die Verfahren am weitesten entwickelt. Bezüglich der geometrischen Modellbildung haben sich verschiedene Techniken in der Computergraphik etabliert. Die vorliegende Aufgabenstellung legt zwei Methoden nahe:

- die Verwendung von geometrischen Modellen auf polygonaler Basis (Polyedermodelle). Vorteilhaft ist die hohe Verarbeitungsgeschwindigkeit aufgrund der Optimierung üblicher Rechnerhardware auf schnelle Polygonausgabe. Aufgrund der Möglichkeit zur direkten Manipulation der rechnerinternen Geometrie-Repräsentation sind Methoden zur Modellmodifikation vergleichsweise wenig aufwendig.
- ein Ansatz auf Basis von Freiformflächen. Sie zeichnen sich durch eine gute visuelle Qualität mit Anpassung der Genauigkeit aus, die glatte Darstellung kommt der Modellierung deformierbarer Objekte entgegen. Außerdem ist eine einfache und realitätsnahe Formveränderung aufgrund der lokalen Einwirkung der Kontrollpunkte möglich. In

der graphischen Datenverarbeitung haben sich die NURBS[5] [Far92, Pie91] durchgesetzt, die auf B-Splines basieren.

In Kapitel 5 werden beide Möglichkeiten behandelt. Durch die prinzipielle Trennung von geometrischer und mechanischer Modellbildung läßt sich auch ein hybrider Ansatz verwirklichen, also die wahlweise Darstellung deformierbarer Objekte als Polyeder- oder Freiformflächenmodell.

Weitere Darstellungsoptionen werden von der Rechnerhardware bestimmt. Die für diese Arbeit zur Verfügung stehende Graphik-Workstation (Silicon Graphics 'Onyx') erlaubte den effizienten echtzeitfähigen Einsatz von Gouraud-Schattierung mit verschiedenen Materialattributen, die Definition verschiedener Lichtquellen und den Einsatz von Texturen. Hierdurch kann der visuelle Realismus der Simulation beträchtlich gesteigert werden.

Bei der *physikalischen Repräsentation* zeigen die Techniken der 'nodalen Systeme' den vielversprechendsten Ansatz, insbesondere hinsichtlich der Echtzeitanforderung. Diese Methode wird deshalb aufgegriffen und in Kapitel 3 zum MESD-Verfahren modifiziert und ausgebaut.

Das Teilgebiet der *Interaktionen* wird bezüglich deformierbarer Körper in der Literatur kaum behandelt. Es existieren zwar schon eine Vielzahl von Verfahren der Kollisionserkennung, insbesondere aus der Robotik. Diese genügen jedoch nicht dem Echtzeitanspruch oder sind zu grob. Außerdem sind die dortigen Ansätze für deformierbare Objekte meist nicht praktikabel, da sie für starre, unveränderliche Geometrien ausgelegt sind. Für die vorliegende Aufgabenstellung wurden deshalb neue Verfahren der Kollisionserkennung und -behandlung sowie der Modellmanipulation entwickelt. In Kapitel 4 werden die erarbeiteten Methoden vorgestellt.

Sehr wichtig ist auch ein geeignetes Schnittstellenkonzept zur Verbindung der Teilsysteme. Durch einen streng modularen Aufbau mit definiertem Datenaustausch und beschränkten Datenmanipulations-Möglichkeiten wird ein strukturiertes System gebildet, das den Prinzipien einer objektorientierten Methodik entspricht (Bild 2.3). Außerdem wird durch diese Systematik die Parallelisierung erleichtert (Kapitel 3.4.3).

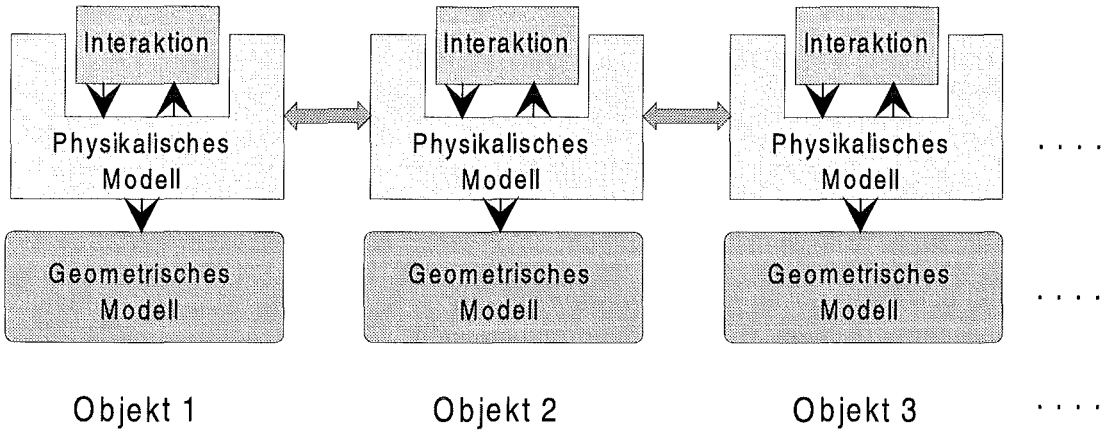


Bild 2.3: Modulare Struktur der Modellierung und Simulation

[5] Non-Uniform Rational B-Spline Surface

2.4 Simulationssystem 'KISMET'

Für die Einsatzplanung, Simulation, Programmierung und Überwachung von Handhabungsgeräten und Robotern wurde am Forschungszentrum Karlsruhe das Softwarepaket '**KISMET**' (**K**inematic **S**imulation, **M**onitoring and Off-Line Programming **E**nvironment for **T**elerobotics) entwickelt [Kue91]. Das System soll Bediener und Planer von Handhabungseinrichtungen durch eine möglichst realitätsgetreue und aufgabengerechte Szenenpräsentation während der Vorbereitung und Ausführung von fernbedienten Aufgaben unterstützen. KISMET basiert auf geometrischen, kinematischen und dynamischen Modellen der Arbeitsumgebung und der Handhabungsgeräte, mit denen dreidimensionale, synthetische Ansichten in Echtzeit erzeugt werden ('künstliches Sehen').

Die Definition, der Import und die Umsetzung verschiedenster Geometrieformate ist möglich. Die Datenstruktur der Modelle ist hierarchisiert, damit auch verschiedene Detaillierungsstufen effizient darstellbar sind. Die Fähigkeiten der kinematischen Modellierung sind sehr umfangreich, verschiedene Licht- und Schattierungsmodelle erlauben eine Anpassung an die spezifischen Anforderungen. Weitere Funktionalitäten wie beispielsweise Skript-Befehle[6], Monitoring-Modus[7] und Interprozeß-Kommunikation ermöglichen eine effektive Unterstützung und Steuerung der Simulation.

Aufgrund dieser Eigenschaften wird KISMET als Basissystem der vorgestellten Simulationsumgebung verwendet. Die in dieser Arbeit entwickelten Verfahren wurden in KISMET integriert. Hierbei konnten vorhandene Softwaremodule zur Simulationssteuerung, kinematischen Berechnung der Instrumente und Graphikverarbeitung übernommen werden, die graphische Bedienoberfläche wurde leicht modifiziert. Die Implementierung erfolgte mit der Programmiersprache C unter Verwendung der Graphikbibliothek GL[8] [SGI91], als Hardwareplattform dienten Hochleistungs-Graphikworkstations der Firma 'Silicon Graphics' mit dem Betriebssystem UNIX[9].

[6] Textuelle Schnittstelle zum Kommandoprozessor.
[7] Überwachungsbetrieb: realitätsgetreue 'On-Line'-Darstellung der momentanen, tatsächlichen Bewegungen und Zustände eines technischen Systems.
[8] 'Graphics Library' der Firma Silicon Graphics, wird abgelöst durch die systemübergreifende Graphik-Bibliothek 'OpenGL'.
[9] Das UNIX-Derivat der Firma Silicon Graphics trägt die Bezeichnung 'IRIX'.

3 Das MESD-Verfahren zur physikalischen Repräsentation deformierbarer Objekte

Im 'Stand der Technik' wurden Verfahren zur physikalischen Repräsentation beschrieben, die auf nodalen Netzen basieren. Dieser Ansatz wird aufgegriffen und im Rahmen dieser Arbeit zum MESD-Verfahren (Methodik der **M**odellbildung zur **e**chtzeitfähigen **S**imulation und **M**anipulation **d**eformierbarer Objekte über physikalisch-basierte nodale Systeme) weiterentwickelt. In diesem Kapitel sollen die Anforderungen spezifiziert und das MESD-Verfahren vorgestellt werden.

3.1 Anforderung und Grundlagen

Zuerst gilt die Frage zu klären, welche Eigenschaften eine physikalische Repräsentation zu erfüllen hat.

Definition 3.1:

Das Ziel der **physikalischen Repräsentation** ist die realitätsgetreue Simulation des Verhaltens deformierbarer Körper. Grundlage hierfür ist die Erstellung eines Modells, in dem die Eigenschaften eines Objektes mit Hilfe von physikalischen Gesetzmäßigkeiten beschrieben werden.

Im Rahmen dieser Arbeit wird die physikalische Repräsentation auf das Teilgebiet der Mechanik beschränkt. Die Kinematik und Kinetik beschreibt hierbei die Bewegung eines massebehafteten Körpers unter Einfluß von (äußeren und inneren) Kräften. Zentrale Aufgabe ist die Bestimmung und Auswertung der Bewegungsgleichungen des Objektes.

3.1.1 Grundlagen aus der Mechanik

In diesem Unterkapitel sollen die für das Verständnis der Methode wichtigen mechanischen Grundlagen kurz angesprochen werden. Auf eine tiefgreifende Einführung und Herleitung der Gleichungen wird an dieser Stelle verzichtet, interessierte Leser werden auf [Spi76, KW91, Deu96] verwiesen.

Grundlegend ist die Aussage, wie sich eine Verformung, deren Ursachen und Auswirkungen mathematisch beschreiben lassen. Elementar hierbei ist zunächst der Begriff eines 'deformierbaren Körpers' und dessen physikalische Definition.

Definition 3.2:

Ein (dreidimensionaler) **Körper** ist eine zusammenhängende, massebehaftete Punktemenge $P \subset \mathbb{R}^3$ mit infinitesimaler Punktausdehnung. Die Gesamtheit aller dieser Volumenelemente formt die Geometrie des Körpers, auch beschreibbar als kontinuierliche Ansammlung von Masseteilchen.

Ein einfach zu beschreibender Sonderfall ist der Starrkörper. Dieser zeichnet sich dadurch aus, daß seine Geometrie zu jeder Zeit unveränderlich ist. Die Abstände der Punkte innerhalb der Menge *P* bleiben also bei allen Einwirkungen konstant.

Bemerkung 3.1:

Für die statische Zustandsbeschreibung **x** eines ***Starrkörpers*** *S* im dreidimensionalen Raum reichen sechs Zustandsvariablen aus: drei translatorische Freiheitsgrade (**r**) sowie drei rotatorische Freiheitsgrade (φ).

$$\mathbf{x} = [\mathbf{r}, \varphi]^T = [r_x, r_y, r_z, \varphi_x, \varphi_y, \varphi_z]^T \qquad (3.1)$$

Zugrundegelegt wird die Euklidsche Geometrie [Spi76] im kartesischen Koordinatensystem mit frei definiertem Ursprungspunkt (Bild 3.1). Für eine vollständige dynamische Beschreibung werden zusätzlich noch Angaben über die Geschwindigkeiten ($\dot{\mathbf{r}}, \dot{\varphi}$) sowie die Beschleunigungen ($\ddot{\mathbf{r}}, \ddot{\varphi}$) des Starrkörpers benötigt.

Definition 3.3:

Ein ***deformierbarer Körper*** *D* ist ein allgemeiner Körper im Sinne von Definition 3.2. Im Gegensatz zum Starrkörper müssen die Punktabstände nicht zu jeder Zeit konstant bleiben.

Die Anzahl der Freiheitsgrade und damit die zur vollständigen Beschreibung benötigten Parameter wachsen hierbei stark an - bis hin zu einer unendlichen Zahl von Zustandsparametern, wenn man obige Definitionen zugrundelegt. Deshalb werden vereinfachte Beschreibungsformen verwendet, die das Verhalten deformierbarer Körper allgemein charakterisieren.

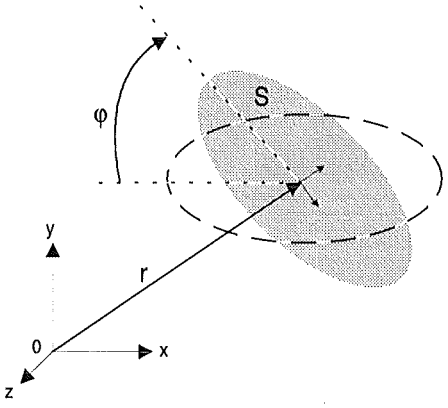


Bild 3.1: Zustandsbeschreibung eines Starrkörpers

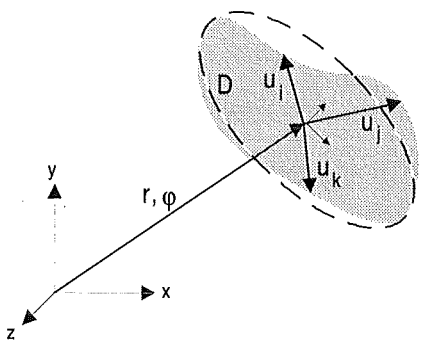


Bild 3.2: Superpositionsprinzip: Deformierbarer Körper

Hilfreich ist die Verwendung des Superpositionsprinzips. Die Bewegung eines deformierbaren Körpers wird in eine globale Starrkörperbewegung und in eine überlagerte, lokale Verformung unterteilt (Bild 3.2). Durch diese Zerlegung werden die physikalischen Gesetzmäßigkeiten leichter handhabbar. Die Menge der Ortsvektoren **u** aller Körperpunkte bestimmt die Gestalt des deformierbaren Körpers. Für eine vollständige dynamische Beschreibung sind wiederum - analog zum Starrkörper - Informationen über die Geschwindigkeiten und Beschleunigungen der Punktvektoren notwendig.

Durch Koordinatensystemwechsel auf das lokale System des deformierbaren Körpers kann die globale Bewegung in den folgenden Betrachtungen unberücksichtigt bleiben. Das Verhalten deformierbarer Körper kann mit Hilfe der Elastizitätstheorie [KW91] beschrieben werden, die eine Definition und Verknüpfung von Zustandsgrößen und Materialparametern zur physikalischen Beschreibung von Verformungen bereitstellt[1].

Gegenstand der Elastizitätslehre ist die Beschreibung des Zusammenhangs zwischen den Verschiebungen der Körperpunkte (**u**) sowie der inneren ($\mathbf{F}_i$) und äußeren ($\mathbf{F}_a$) Kräfte, die an den Körperpunkten angreifen (Bild 3.3). Bei den nachfolgenden Definitionen wird ein Körperpunkt stets als infinitesimal kleines Volumenelement (dV) betrachtet.

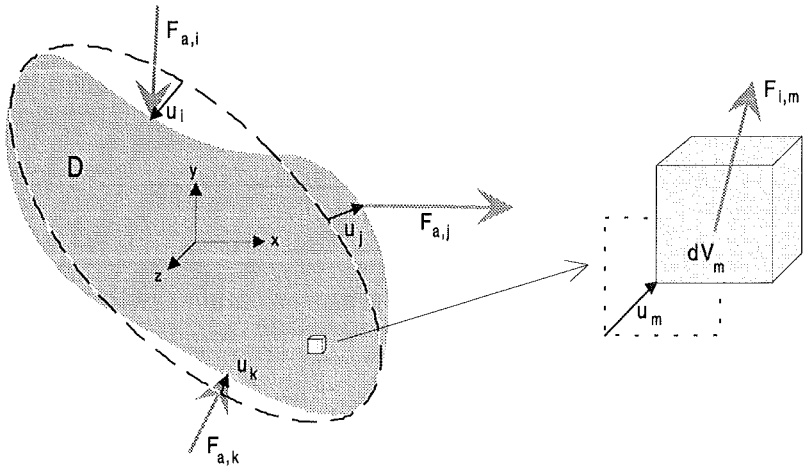


Bild 3.3: Einwirkung von äußeren Kräften am deformierbaren Körper und Volumenelement

Definition 3.4:

Der ***Verzerrungsvektor*** ε ist definiert als partielle räumliche Ableitung des Verschiebungsvektors **u**. Mit Hilfe eines Differentialoperators $\mathbf{D}_\varepsilon$ ergibt sich folgender Zusammenhang:

$$\varepsilon = (\varepsilon_x, \varepsilon_y, \varepsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{xz})^T = \underbrace{\begin{pmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{pmatrix}}_{\mathbf{D}_\varepsilon} \cdot \begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix} \tag{3.2}$$

[1] An dieser Stelle werden homogene und isotrope Materialien vorausgesetzt, die ein lineares elastisches Verhalten zeigen (Elastizitätsgesetz). Betrachtet wird der allgemeine dreidimensionale Fall.

Die Verzerrungen ε werden als Dehnungen bezeichnet, die γ als Schub- oder Scherverzerrungen. Gleichung (3.2) repräsentiert somit die Verformung eines Volumenelementes. Eine weitere Größe charakterisiert die Spannungsverteilung im Körper, die von der am Volumenelement anliegenden Kraft verursacht wird.

Definition 3.5:

Die ***Spannung*** σ ist definiert als an einem Volumenelement anliegende Kraft **F** pro Flächeneinheit A. Die Kraft kann in eine zur Normalen von A parallele Teilkraft F_n (Normalkraft) und eine dazu senkrecht stehende Teilkraft F_t (Tangentialkraft) zergliedert werden. Man unterscheidet deshalb auch die Normalspannung σ sowie die Tangential-, Scher- oder Schubspannung τ.

$$\sigma = \lim_{\Delta A \to 0} \frac{\Delta F_n}{\Delta A}, \qquad \tau = \lim_{\Delta A \to 0} \frac{\Delta F_t}{\Delta A} \tag{3.3}$$

$$\sigma = (\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{yz}, \tau_{xz})^T = \begin{pmatrix} \frac{\partial}{\partial y \partial z} & 0 & 0 \\ 0 & \frac{\partial}{\partial x \partial z} & 0 \\ 0 & 0 & \frac{\partial}{\partial x \partial y} \\ \frac{\partial}{\partial x \partial y} & \frac{\partial}{\partial x \partial y} & 0 \\ 0 & \frac{\partial}{\partial y \partial z} & \frac{\partial}{\partial y \partial z} \\ \frac{\partial}{\partial x \partial z} & 0 & \frac{\partial}{\partial x \partial z} \end{pmatrix} \cdot \begin{pmatrix} F_x \\ F_y \\ F_z \end{pmatrix} \tag{3.4}$$

Ein materialabhängiges Gesetz definiert nun die Beziehung zwischen Verzerrung und Spannung. Die spezifischen Stoffeigenschaften bestimmen somit, wie sich die Spannungen auf die Verzerrungen auswirken und umgekehrt. In dem vorausgesetzten Fall der linearen, isotropen Elastizität kommt das Hooksche Gesetz zum Einsatz:

$$\begin{pmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{xz} \end{pmatrix} = \underbrace{\frac{1}{E} \begin{pmatrix} 1 & -\upsilon & -\upsilon & 0 & 0 & 0 \\ -\upsilon & 1 & -\upsilon & 0 & 0 & 0 \\ -\upsilon & -\upsilon & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2(1+\upsilon) & 0 & 0 \\ 0 & 0 & 0 & 0 & 2(1+\upsilon) & 0 \\ 0 & 0 & 0 & 0 & 0 & 2(1+\upsilon) \end{pmatrix}}_{\mathbf{C}} \cdot \begin{pmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{xz} \end{pmatrix} \tag{3.5}$$

oder kurz: $\varepsilon = \mathbf{C} \cdot \sigma$

Das Elastizitätsmodul E und die Poissonzahl ν sind materialabhängige Konstanten. Zur Bestimmung des mechanischen Verhaltens müssen Randbedingungen berücksichtigt werden, wie geometrische Beschränkungen, vorgegebene Verschiebungswerte und eingeprägte Kräfte. Für ein Teilvolumen des Körpers D ergibt im statischen Kräftegleichgewicht weiterhin die Gleichgewichtsbedingungen (**p** ist der Belastungsvektor für Volumenlasten):

$$\mathbf{D}_\varepsilon^T \cdot \sigma - \mathbf{p} = 0 \qquad \textit{mit} \quad \mathbf{F} = \iiint_V \mathbf{p}\, dV \tag{3.6}$$

Die Gleichungen (3.2), (3.4), (3.5) und (3.6) ergeben zusammen mit den Randbedingungen ein System von partiellen Differentialgleichungen 2. Ordnung, welches das elastische

Verhalten des Körpers beschreibt. Eine direkte analytische Lösung dieser Gleichungen ist nur in den einfachsten Fällen möglich. Ansonsten muß eine Abstrahierung des Körpers vorgenommen werden - eine Zerlegung des Objektes in einzelne, endliche Teilelemente mit genau definierbaren Eigenschaften. Dies führt direkt zu der Finite-Elemente-Methodik - und, in einem höheren Abstraktionsniveau, zum MESD-Verfahren.

3.1.2 Grundlagen nodaler Netze, Abgrenzung zur FEM

Eine mathematisch exakte Repräsentation des Verhaltens deformierbarer Körper ist meist nicht möglich. Ziel ist deshalb eine vereinfachende Darstellung der physikalischen Gesetzmäßigkeiten, so daß eine Simulation in Echtzeit möglich ist. Es wird deshalb ausgehend von den Definitionen in Kapitel 3.1.1 eine Beschreibung gesucht, die eine gute Näherung des Verhaltens ermöglicht und gleichzeitig die Komplexität der zugrundeliegenden Mathematik stark reduziert. Der erste Schritt ist somit die Aufstellung eines passenden Modells - die Modellbildung.

Ausgehend von der Beschreibung des Körpers D durch Volumenelemente dV kann ein Körper äquivalent zu Definition 3.2 durch die Gleichung (3.7) beschrieben werden. Für eine dynamische Beschreibung ist die Berücksichtigung der Masse notwendig. Es muß daher vorausgesetzt werden, daß die Volumenelemente massebehaftet sind (Massendichte ρ). Die Gesamtmasse m_D des Körpers D resultiert aus Gleichung (3.8).

$$D: \quad V_D = \iiint_{(P)} dV, \qquad m_D = \iiint_{(P)} \rho \, dV \qquad (3.7), (3.8)$$

Eine Anzahl von infinitesimalen Volumenelementen aus der gesamten Punktmenge P des Körpers kann in Gruppen zusammengefaßt werden. Hierdurch entstehen Teilvolumen V_i mit entsprechenden endlichen, nichtverschwindenden Massen m_i. Wird das gesamte Körpervolumen in n Teilvolumen aufgeteilt, so erhält man eine approximierte Darstellung des Körpers D (Bild 3.4). Wichtig ist hierbei, möglichst Teilbereiche des Körpers zusammenzufassen, die gleiche Eigenschaften besitzen und damit gemeinsam beschrieben werden können. Diese entstehenden 'Finiten Elemente' repräsentieren den deformierbaren Körper, der nunmehr als System mehrerer einzelner Teilkörper angesehen werden kann.

Aufgrund dieses Vorgehens ist ein Übergang von der kontinuierlichen Darstellung in eine diskrete Beschreibung des Körpers D notwendig. Aus Gleichung (3.7) ergibt sich (3.9), aus (3.8) resultiert (3.10).

$$D: \qquad V_D = \sum_n V_i, \qquad m_D = \sum_n m_i \qquad (3.9), (3.10)$$

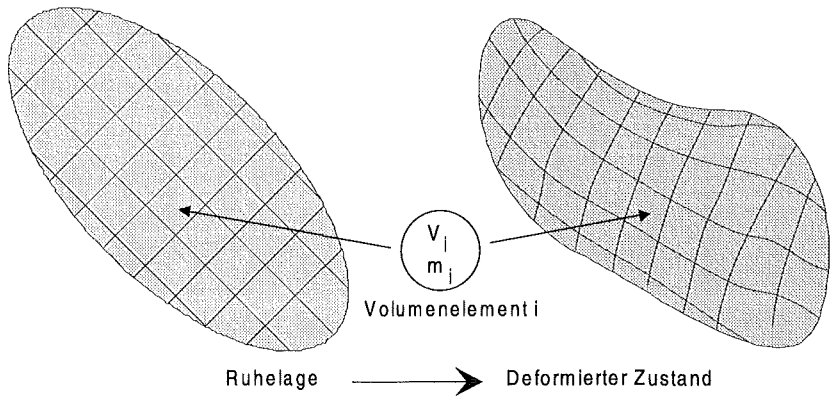


Bild 3.4: Schnittdarstellung eines deformierbaren Körpers mit Volumenelementen

Zur Lösung der Bewegungsgleichungen für ein deformierbares Objekt wird das komplexe Problem durch diese Vorgehensweise in viele, aber kleinere Teilprobleme zerlegt. Dies ergibt natürlich nur Sinn, wenn die verwendeten Elemente physikalisch leichter zu beschreiben und mathematisch leichter auszuwerten sind als das Gesamtsystem. In der klassischen Finite-Elemente-Methodik verwendet man deshalb meist geometrisch einfache Strukturen wie Tetraeder oder Quader[2], deren mechanische Eigenschaften sich mit geschlossenen Formeln charakterisieren lassen. Die Eckpunkte der Elemente werden als Knoten bezeichnet.

Die Randflächen dieser Elemente werden durch Formfunktionen [KW91] beschrieben. Diese können im einfachsten Fall linear sein (Bild 3.5), hiermit lassen sich C^0-stetige Übergänge erzielen. Für bessere Approximationen werden Ansatzfunktionen höheren Grades verwendet, die Übergänge höherer Ordnung ermöglichen (C^1-, C^2-stetig) und folglich aus gekrümmten Randflächen bestehen.

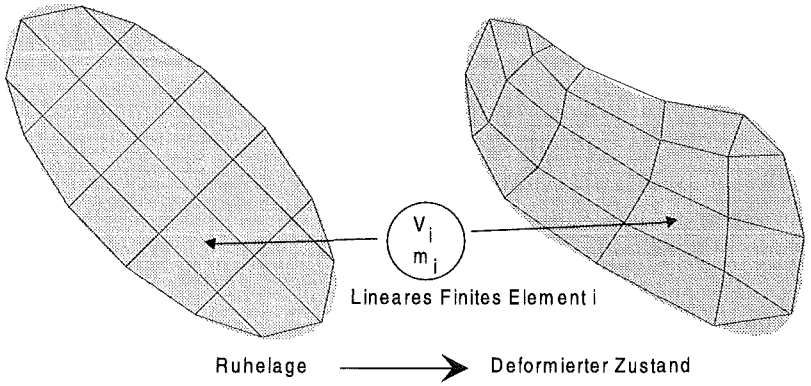


Bild 3.5: Deformierbarer Körper, approximiert mit linearen Elementen

Durch die Zerlegung soll die Konsistenz und Stetigkeit des Körpers gewährleistet bleiben. Deshalb sind neben den (eingeprägten) äußeren Randbedingungen auch Übergangsbedingungen zwischen den einzelnen Elementen zu berücksichtigen. Diese Randbedingungen bewirken auch eine Verknüpfung der Elemente und eine mathematische Kopplung. Die Komplexität der resultierenden Gleichungssysteme ist daher sehr hoch, die Dimensionen der Lösungsmatrizen betragen im dreidimensionalen Fall (3n x 3n), wobei n die Anzahl der Knoten im FE-Modell ist. Der Aufwand zur allgemeinen Lösung eines solchen Gleichungssystems besitzt die Komplexität $O(3n^3)$. Eine direkte Auswertung unter Echtzeitbedingungen ist hierbei nur bei sehr kleinen Modellgrößen zu realisieren[3].

Von besonderer Bedeutung innerhalb dieser Arbeit ist die Abgrenzung der FEM zu den nodalen Systemen. An dieser Stelle soll die Finite-Elemente-Methodik deshalb nicht näher erläutert werden, sie wird in Standardwerken wie beispielsweise [BW76], [ZM83], [KW91] ausführlich beschrieben.

Eine weitere Abstraktionsmöglichkeit besteht in der Diskretisierung der Volumenelemente auf nulldimensionale Knoten. Durch die Konzentration der Masse werden keine kontinuierlichen Elemente betrachtet, sondern diskrete Punktelemente. Die Beziehung zwischen diesen Punktelementen muß durch weitere Strukturen aufrechterhalten werden, die das elastische Verhalten einbringen. Im einfachen linearen Fall können dies beispielsweise virtuelle Federn sein, die damit die Elastizität charakterisieren.

[2] Bei räumlichen Problemstellungen. Viele dreidimensionale Probleme lassen sich jedoch auf zweidimensionale oder gar eindimensionale Fälle zurückführen, so daß meist Scheiben- oder Plattenelemente bzw. Balken- oder Stabelemente verwendet werden.

[3] Die Betonung liegt hier auf dem allgemeinen Lösungsfall. In vielen Fällen handelt es sich um 'lichte' Matrizen (viele Nullelemente), hierzu gibt es effizientere Auswertungsverfahren. Die Grundaussage bleibt jedoch bestehen.

Definition 3.6:

Ein ***nodales System*** N mit linearelastischen Eigenschaften ist definiert durch:

- eine Menge von Punkten mit definierter endlicher Masse m_K und verschwindenden Volumen ($\Delta V \rightarrow 0$), die als Knoten K bezeichnet werden.
- eine Menge von masselosen Federelementen E, die zwischen jeweils zwei Knoten aufgespannt werden und auf diese eine Kraft ausüben können.

$$D: \quad N = \{K, E\} \tag{3.11}$$

Der deformierbare Körper D wird somit durch ein nodales Netz repräsentiert.

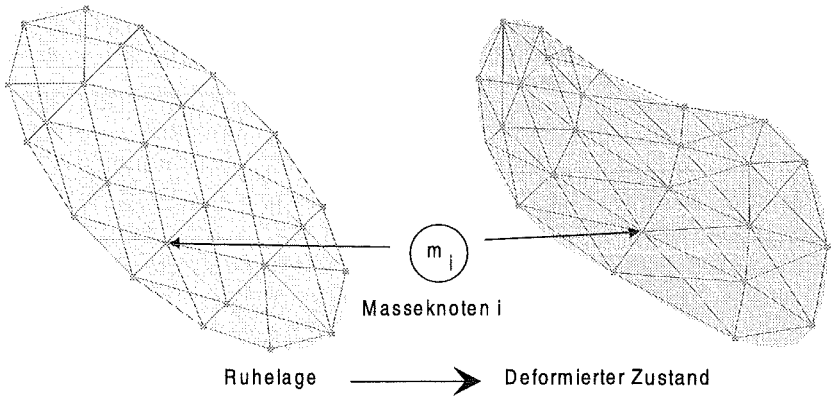


Bild 3.6: Deformierbarer Körper, approximiert mit nodalem System

Der große Vorteil hinsichtlich der Auswertung der Bewegungsgleichungen ist der Wegfall der Formfunktionen und damit der Übergangsbedingungen zwischen den einzelnen Elementen des deformierbaren Körpers. Damit sind die physikalischen Sachverhalte durch die einfachere Mathematik der Massepunktsysteme beschreibbar und somit für eine echtzeitfähige Simulation besser geeignet. Die Exaktheit der Repräsentation des physikalischen Verhaltens wird durch diese relativ grobe Approximation jedoch reduziert.

Das Prinzip der nodalen Systeme wird hinsichtlich den Anforderungen dieser Arbeit erweitert und zum MESD-Verfahren ausgebaut.

3.2 Konzept des MESD-Verfahrens und Modellbildung

Nodale Systeme werden auch als 'physikalisch-basiert' bezeichnet, da sich grundlegende Gesetzmäßigkeiten der Physik, insbesondere die Massepunkt- und Starrkörperdynamik, auf deformierbare Objekte übertragen lassen.

Definition 3.7:

Das **MESD-Verfahren** (*Methodik der Modellbildung zur echtzeitfähigen Simulation und Manipulation deformierbarer Objekte über physikalisch-basierte nodale Systeme*) beruht auf nodalen Netzen, die auch als vereinfachte Finite-Elemente-Systeme betrachtetet werden können. Ein Körper wird durch eine Menge von attributbehafteten Masseknoten K und durch eine Menge von verallgemeinerten, kraftausübenden Verbindungselementen V repräsentiert.

$$D: \quad M = (K, V) \tag{3.12}$$

In Ergänzung zu den allgemeinen Anforderungen der physikalischen Modellbildung (Kapitel 2.2.2) sollen die spezifischen Ansprüche an das MESD-Verfahren kurz dargelegt werden.

- Durch den nodalen Ansatz wird das resultierende mathematische Modell relativ einfach und überschaubar, so daß mit entsprechenden numerischen Verfahren eine Simulation in Echtzeit möglich wird. Um die Stabilität zu gewährleisten, müssen jedoch erweiterte Modellbildungsmethoden einbezogen werden.
- Die Genauigkeit der Simulation ist sicherlich durch die grobe Approximation beschränkt. Dennoch kann die Güte der Modellbildung durch die Anzahl der verwendeten Knoten und Verbindungselemente an die spezifischen Anforderungen angepaßt und gesteuert werden.
- Die Modellierungsfreiheit des Körpers ist mit dem nodalen Ansatz gewährleistet, die Massenverteilung kann frei festgelegt werden. Zur Berücksichtigung globaler Bewegungen muß die Charakteristik der Masseknoten verallgemeinert werden. Dies bedingt die Einbeziehung weiterer Attribute neben der Masse, die das dynamische Verhalten spezifizieren. Die äußere Form läßt sich in Abhängigkeit der Modellgröße beliebig genau approximieren. Zur realistischen Simulation des mechanischen Verhaltens sollen die Verbindungselemente nicht auf lineare Federn beschränkt bleiben, sondern prinzipiell beliebige nichtlineare Charakteristiken besitzen können. Die Modellbildung soll eine Beschränkung der Freiheitsgrade möglichst vermeiden und damit die Simulationsfreiheit gewährleisten.
- Ein wichtiger Aspekt bei der Modellbildung soll die Möglichkeit zur Manipulation der deformierbaren Objekte sein, also die interaktive Modifizierbarkeit des Modells zur Simulationslaufzeit. Bei den nodalen Systemen bedingt dies ein Umorganisieren der Netzstruktur, dem Löschen und Hinzufügen von Knoten und Verbindungselementen und eine Anpassung der Parameter. Das MESD-Verfahren und die zugrundeliegende Datenstruktur muß dieser wichtigen Anforderung genügen.

3.2.1 Masseknoten

Definition 3.8:

Ein ***Masseknoten*** im Sinne des MESD-Verfahrens ist ein diskreter Punkt (Partikel) mit spezifizierter Masse innerhalb des zu approximierenden Körpers. Dieser Knoten kann neben der kinematischen Beschreibung weitere Attribute besitzen. Eine Menge K von Knoten repräsentiert die Massenverteilung des Körpers und bestimmt damit die Dynamik des Objektes. Die Gesamtheit der einzelnen Knoten-Bewegungen repräsentiert die lokalen Verformungen und die globale Bewegung des deformierbaren Körpers.

Ein zentrales Problem der Modellbildung besteht in der Diskretisierung der Objektmasse, also die Bestimmung der Massenverteilung mittels der Masseknoten. Hierbei sind prinzipiell zwei Ansätze möglich:

- Ein Volumenmodell beschreibt das Verhalten als räumliches Objekt, es erfolgt eine homogene Diskretisierung über das gesamte Körpervolumen. Das entstehende Knotennetz besitzt damit eine dreidimensionale Ausdehnung (Knotengitter, Bild 3.6).
- Ein Oberflächenmodell beschreibt das Verhalten eines Objekts über dessen Oberflächendeformation, also der Verformung des sichtbaren Teils der Geometrie. Hierbei wird ein zweidimensionales Knotennetz über den Körper gelegt.

Das Volumenmodell liefert im allgemeinen eine realistischere Aussage über das Verhalten, insbesondere bei Verformungen, die nicht lokal beschränkt sind. Andererseits wird das nodale Netz durch die große Zahl der inneren Knoten sehr umfangreich. Ziel des MESD-Verfahrens ist es daher, mit möglichst wenigen inneren Knoten ein realistisches Simulationsverhalten, auch in räumlicher Hinsicht, zu erzielen. Hierbei sollen neben den lokalen Deformationen auch die globalen Bewegungen und Verformungen berücksichtigt werden. Sinnvoll ist deshalb die Einführung verschiedener Knotentypen, die in dieser Arbeit als G-, Z- und F-Knoten bezeichnet werden.

Die äußeren Knoten befinden sich auf der Oberfläche und repräsentieren damit die geometrische Form des deformierbaren Körpers. Die Gesamtheit dieser sogenannten G-Knoten charakterisiert die Geometrie, sie symbolisieren die äußere Schale des Objektes.

Definition 3.9:

Der ***G-Knoten*** besitzt die mechanische Charakteristik eines *Massepunktes*. Sein kinematisches Verhalten ist durch die Angabe seines relativen Positionsvektors **u**, seiner Geschwindigkeit $\partial\mathbf{u}/\partial t$ sowie seiner Beschleunigung $\partial^2\mathbf{u}/\partial t^2$ vollständig beschrieben. Ein G-Knoten führt somit nur eine translatorische Bewegung relativ zu seinem Bezugskoordinatensystem aus. Der diese Bewegung beschreibende Vektor wird **G-Vektor** genannt.

$$\mathbf{G} = (u_x, u_y, u_z, \dot{u}_x, \dot{u}_y, \dot{u}_z, \ddot{u}_x, \ddot{u}_y, \ddot{u}_z)^T \tag{3.13}$$

Grundlage der physikalischen Beschreibung ist die Newtonsche Mechanik [Spi76]. Das zweite Newtonsche Bewegungsgesetz beschreibt das Verhalten eines Körpers unter Einfluß äußerer Kräfte (**v**: Geschwindigkeitsvektor, **a**: Beschleunigungsvektor):

$$\mathbf{F} = m\frac{d\mathbf{v}}{dt} = m \cdot \dot{\mathbf{v}} = m \cdot \mathbf{a} \tag{3.14}$$

Hierbei ist das Superpositionsprinzip zugrundegelegt. Wirken auf einen Punkt mehrere Kräfte F_1, F_2 ... F_n, so addieren sich die Kraftvektoren zu einer resultierenden Kraft F.

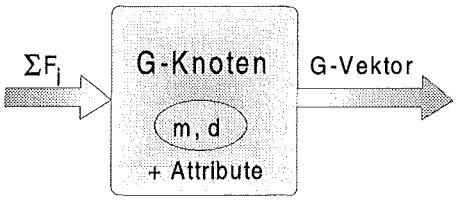


Bild 3.7: Blockschema eines G-Knotens

$$\mathbf{F} = \sum_{i=1}^{n} \mathbf{F}_i \tag{3.15}$$

Die Geschwindigkeit eines Körpers ist definiert als die zeitliche Ableitung des Positionsvektors:

$$\mathbf{v} = \frac{d\mathbf{u}}{dt} = \dot{\mathbf{u}} \tag{3.16}$$

Als resultierende Differentialgleichung für einen Knoten ergibt sich im allgemeinen Fall mit der Dämpfungsfunktion $d(\mathbf{u}, \dot{\mathbf{u}})$, der inneren Kraftfunktion $f(\mathbf{u}, \dot{\mathbf{u}})$ und der äußeren Kraft $\mathbf{F}_A$:

$$m \cdot \ddot{\mathbf{u}} + d(\mathbf{u}, \dot{\mathbf{u}}) + f(\mathbf{u}, \dot{\mathbf{u}}) = \mathbf{F}_A \tag{3.17}$$

Setzt man eine lineare, viskose Dämpfung $d(\mathbf{u}, \dot{\mathbf{u}}) = d \cdot \dot{\mathbf{u}}$ (Dämpfungskonstante d) sowie eine lineare innere Rückstellkraft $f(\mathbf{u}, \dot{\mathbf{u}}) = k \cdot \mathbf{u}$ (Federkonstante k) voraus, so vereinfacht sich Gleichung (3.17) auf eine lineare Differentialgleichung 2. Ordnung mit konstanten Koeffizienten, die mathematisch das Verhalten eines gedämpften Masse-Feder-Pendels beschreibt.

$$m \cdot \ddot{\mathbf{u}} + d \cdot \dot{\mathbf{u}} + k \cdot \mathbf{u} = \mathbf{F}_A \tag{3.18}$$

Als weiterer Knotentyp werden sogenannte Zentral-Knoten eingeführt, die die globale Bewegung des deformierbaren Objektes charakterisieren. Hierzu wird eine überlagerte Starrkörperbewegung angenommen. Aber auch für größere lokale Deformationen sind diese Z-Knoten als Beschreibungsmittel notwendig (Kapitel 3.3).

Definition 3.10:

Der ***Z-Knoten*** besitzt die mechanische Charakteristik eines *Starrkörpers,* dessen Massenschwerpunkt im Knoten liegt. Für die vollständige Beschreibung seines kinematischen Verhaltens ist neben dem G-Vektor zusätzlich die Angabe seines relativen Winkelvektors ϕ, seiner Winkelgeschwindigkeit $\partial\phi/\partial t$ sowie seiner Winkelbeschleunigung $\partial^2\phi/\partial t^2$ notwendig. Ein Z-Knoten kann somit sowohl eine translatorische als auch eine rotatorische Bewegung relativ zu seinem Bezugskoordinatensystem ausführen. Der diese Bewegung beschreibende Vektor wird **Z-Vektor** genannt.

$$\mathbf{Z} = (u_x, u_y, u_z, \dot{u}_x, \dot{u}_y, \dot{u}_z, \ddot{u}_x, \ddot{u}_y, \ddot{u}_z, \phi_x, \phi_y, \phi_z, \dot{\phi}_x, \dot{\phi}_y, \dot{\phi}_z, \ddot{\phi}_x, \ddot{\phi}_y, \ddot{\phi}_z)^T \qquad (3.19)$$

Neben den translatorischen Bewegungsgleichungen (3.13-3.17) sind somit weitere rotatorische Bewegungsgleichungen zu definieren. Entsprechend der zweiten Newtonschen Bewegungsgleichung ergibt sich für die Rotation (**M**: Momentenvektor, ω: Winkelgeschwindigkeitsvektor):

$$\mathbf{M} = \boldsymbol{J}\frac{d\omega}{dt} = \boldsymbol{J}\cdot\dot{\omega} \qquad (3.20)$$

Im Gegensatz zur skalaren Masse m bei der translatorischen Bewegung ist $\boldsymbol{J}$ eine Matrix, die auf der Hauptdiagonalen die jeweiligen Trägheitsmomente in x-, y- und z-Richtung enthält. Hierbei wird jedoch angenommen, daß der Knoten auf den Hauptträgheitsachsen liegt. Deswegen verschwinden die Momente außerhalb der Matrix-Hauptdiagonalen (Deviationsmomente), aus dem allgemeinen Trägheitstensor ergibt sich die Diagonalmatrix $\boldsymbol{J}$ der Hauptträgheitsmomente.

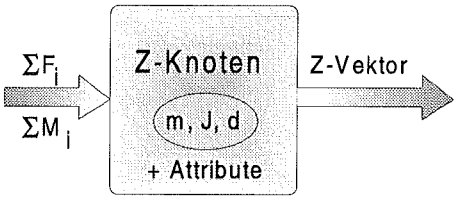


Bild 3.8: Blockschema eines Z-Knotens

Der Momentenvektor **M** ist definiert als Summe der Kreuzprodukte der wirkenden Kraftvektoren mit den jeweiligen Ortsvektoren **r**. Der Ortsvektor (Hebelarm) ist der Vektor vom Angriffspunkt der Kraft zum Knoten.

$$\mathbf{M} = \sum_{i=1}^{n}\mathbf{M}_i = \sum_{i=1}^{n}(\mathbf{r}_i \times \mathbf{F}_i) \qquad (3.21)$$

Die Winkelgeschwindigkeit eines Körpers wird bestimmt aus der zeitlichen Ableitung des Winkelvektors:

$$\omega = \frac{d\phi}{dt} = \dot{\phi} \qquad (3.22)$$

Als beschreibende Differentialgleichung für die rotatorische Bewegung eines Knotens ergibt sich im allgemeinen Fall mit der Dämpfungsfunktion $d(\phi,\dot{\phi})$, der inneren Momentenfunktion $f(\phi,\dot{\phi})$ und dem äußeren Momentenvektor $\mathbf{M_A}$:

$$\boldsymbol{J}\cdot\ddot{\phi} + d(\phi,\dot{\phi}) + f(\phi,\dot{\phi}) = \mathbf{M_A} \qquad (3.23)$$

Als dritter Knotentyp wird der sogenannte F-Knoten definiert. Dieser 'freie' Knoten dient zur Verknüpfung innerhalb des deformierbaren Körpers und damit zur Modellierung des internen, räumlichen Verhaltens.

Definition 3.11:

Der ***F-Knoten*** besitzt die mechanische Charakteristik eines Massenpunktes, hat jedoch im Gegensatz zum G-Knoten keinen direkten Geometriebezug. Für die vollständige Beschreibung seines kinematischen Verhaltens ist der G-Vektor ausreichend.

Die Menge K aller Knoten beinhaltet die Menge aller G-Knoten, Z-Knoten und F-Knoten:

$$K = (G \cup Z \cup F) \tag{3.24}$$

Auf die Verwendung der Knotentypen und Beziehungen der Knoten untereinander wird im Kapitel 3.2.3 näher eingegangen. Die Definitionen innerhalb von KISMET sind im Anhang B aufgeführt.

3.2.2 Verbindungselemente

Definition 3.12:

Ein ***Verbindungselement*** (VE) ist ein allgemeines dynamisches Glied, das zwei Masseknoten k_i, k_j miteinander verknüpft und hierbei auf beide Knoten eine Kraft ausübt. Die Menge aller Verbindungselemente V charakterisiert neben der Knotenmenge das physikalische Verhalten des deformierbaren Körpers.

Zur Definition eines Verbindungselementes werden neben den verknüpften Knoten auch die funktionale Charakteristik sowie VE-Parameter benötigt.

$$v_l = (k_i, k_j, c, par), \qquad v_l \in V \tag{3.25}$$

$$k_i, k_j \in K, \quad c \in \{Funktionscharakteristiken\}, \quad par \in IR^n$$

Nach dem Reaktionsprinzip ('actio gleich reactio') ist die auf die Knoten ausgeübte Kraft vom Betrag gleich, jedoch mit gegensätzlicher Richtung. Als Kraftfunktion für ein Verbindungselement läßt sich somit definieren:

$$\mathbf{F}_i = -\mathbf{F}_j = f_c(\mathbf{G}_i, \mathbf{G}_j, par) \tag{3.26}$$

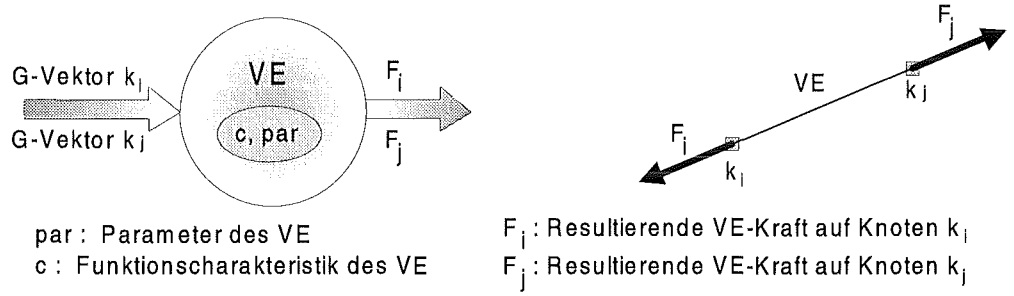


Bild 3.9: Blockschema eines Verbindungselementes, Kraftskizze

Ein Verbindungselement übt im allgemeinen eine Kraft parallel/antiparallel zur Richtung des Verbindungsvektors der beiden verknüpften Knoten aus (Bild 3.9, Positionsdifferenz-Elemente).

Typen und Charakteristiken von Verbindungselementen

Für die kraftbestimmenden Funktionen von Verbindungselementen sind generell beliebige lineare oder auch nichtlineare Funktionen möglich. Die verbundenen Masseknoten müssen das gleiche Bezugskoordinatensystem besitzen, also zum gleichen Objekt gehören. Die VE-Funktion kann in mehrere Teilblöcke unterteilt werden, die Typ und Charakteristik festlegen (Bild 3.9). Die Ausprägungen dieser Teilblöcke sind kombinierbar, so daß eine Vielzahl von VE-Funktionen definierbar ist.

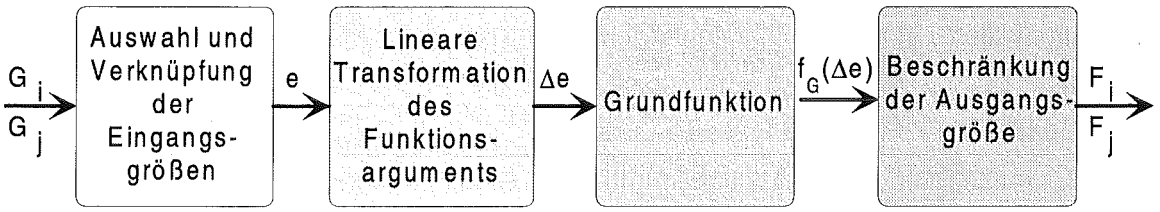


Bild 3.10: Funktionsblöcke der Typdefinition von Verbindungselementen

Komplexere Funktionen können aus mehreren, einzelnen Verbindungselementen nachgebildet werden. Hierzu können mehrere Elemente zwischen zwei Knoten 'parallelgeschaltet' oder auch über einen Zwischenknoten hintereinander geknüpft werden. Eine Analogie mit der Netzwerktheorie in der Elektrotechnik ist hierbei zu erkennen. Eine Einschränkung bezüglich der VE-Funktionen muß jedoch angeführt werden:

Bemerkung 3.2:

Die Funktionen der Verbindungselemente sollten stets C^0-stetig sein, da ansonsten die Systemstabilität nicht gewährleistet werden kann.

Eine C^1-Stetigkeit muß nicht vorausgesetzt werden, da die resultierenden VE-Kräfte nicht mit ihren Ableitungen in die Systemdifferentialgleichung (3.17) eingehen. Im folgenden sollen die Teilblöcke der Verbindungselemente näher betrachtet werden, analog zu Bild 3.9.

Definition 3.13:

Aus den Eingangsgrößen, den G-Vektoren der Knoten, wird die unabhängige Variable e der VE-Funktion bestimmt. Hierbei sind zwei Definitionen sinnvoll, die jeweils aus Differenzenvektoren einen skalaren Eingangswert bilden:

Positionsdifferenz-Element: $$e = \left\| \mathbf{p}_i - \mathbf{p}_j \right\| \tag{3.27}$$

Geschwindigkeitsdifferenz-Element: $$e = \left\| \dot{\mathbf{p}}_i - \dot{\mathbf{p}}_j \right\| \tag{3.28}$$

$\mathbf{p}_i$: Positionsvektor des ersten Knotens

$\mathbf{p}_j$: Positionsvektor des zweiten Knotens

$\dot{\mathbf{p}}_i$: Geschwindigkeitsvektor des ersten Knotens

$\dot{\mathbf{p}}_j$: Geschwindigkeitsvektor des zweiten Knotens

Die Richtung **r** (normierter Vektor) der resultierenden Kraftvektoren ergibt sich aus dem Differenzvektor der beiden betroffenen Masseknoten:

$$\mathbf{r} = \frac{\mathbf{p}_i - \mathbf{p}_j}{\left\|\mathbf{p}_i - \mathbf{p}_j\right\|} \qquad \text{(bei Positionsdifferenz)} \qquad (3.29)$$

$$\mathbf{r} = \frac{\dot{\mathbf{p}}_i - \dot{\mathbf{p}}_j}{\left\|\dot{\mathbf{p}}_i - \dot{\mathbf{p}}_j\right\|} \qquad \text{(bei Geschwindigkeitsdifferenz)} \qquad (3.30)$$

Das Positionsdifferenz-Element bildet die Grundlage für abstandsproportionale Federelemente, während viskose Dämpfungselemente auf Basis der Geschwindigkeitsdifferenz arbeiten.

Definition 3.14:

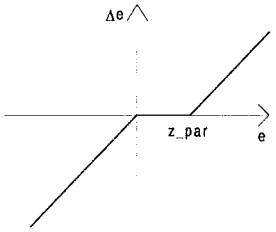
Die lineare Transformation sorgt für eine abschnittsweise Anpassung der Eingangsgröße, insbesondere die Berücksichtigung der Ruhelage (l_0). s_1 ist die obere und s_2 die untere Schranke der abschnittsweisen Transformation, b ein Proportionalitätsfaktor.

$$\Delta e = b \cdot (e - l_0) \qquad \textit{für} \quad s_1 < e \le s_2 \qquad (3.31)$$

Die lineare Transformation ermöglicht insbesondere die Berücksichtigung der Ruhelage, in dem der Offset-Wert l_0 als Ruhelagenauslenkung angenommen wird. Gleichung (3.31) kann für jeden Abschnitt des Eingangswertebereichs von e getrennt spezifiziert werden. Beispiele hierzu sind die Definition von Totzonen, die zur Modellierung von plastischen Verhalten herangezogen werden.

Beispiel: Totzone bei Zugbedingung

$$\Delta e = \begin{cases} e & \textit{für} \quad e \le 0 \\ 0 & \textit{für} \quad 0 < e \le z\_par \\ (e - z\_par) & \textit{für} \quad e > z\_par \end{cases} \qquad (3.32)$$



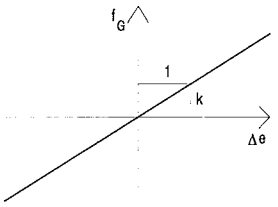
z_par: Schrankenparameter

Die Grundfunktionen f_G sind die Basis für die Verknüpfung der Eingangsvariable Δe mit der resultierenden Ausgabekraft F des Verbindungselementes. Obwohl die verschiedensten nichtlinearen Funktionstypen denkbar wären, sind aus Effizienzgründen vor allem zwei Charakteristiken praktikabel: das lineare Element und das quadratische Element.

Definition 3.15:

Ein **lineares Element** ist definiert durch:

$$F = f_G(\Delta e) = k \cdot \Delta e \qquad (3.33)$$

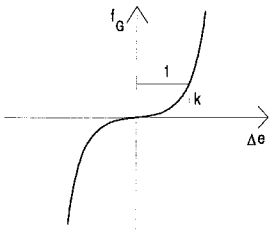


k: Lineare Proportionalitätskonstante
F: Betrag der resultierenden VE-Kraft

Definition 3.16:

Ein **quadratisches Element** ist definiert durch:

$$F = f_G(\Delta e) = \begin{cases} +k \cdot (\Delta e)^2 & \text{für} \quad \Delta e \geq 0 \\ -k \cdot (\Delta e)^2 & \text{für} \quad \Delta e < 0 \end{cases} \tag{3.34}$$

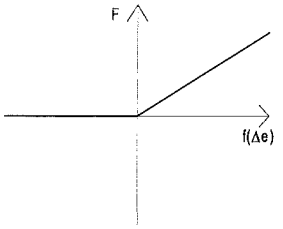


k: Quadratische Proportionalitätskonstante
F: Betrag der resultierenden VE-Kraft

Entsprechend dem letzten Glied der VE-Blockdarstellung kann die resultierende Kraft abschnittsweise beschränkt werden. Beispiele sind Elemente, die nur in einer Belastungsrichtung aktiv sind, sowie absolute Kraftbegrenzungen.

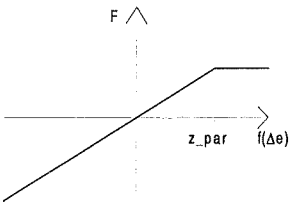
Beispiel: Beschränkung auf Zugbedingungen

$$F = \begin{cases} 0 & \text{für} \quad f(\Delta e) \leq 0 \\ f(\Delta e) & \text{für} \quad f(\Delta e) > 0 \end{cases} \tag{3.35}$$



Beispiel: Kraftschranke bei Zugbedingung

$$F = \begin{cases} f(\Delta e) & \text{für} \quad f(\Delta e) \leq z\_par \\ f(z\_par) & \text{für} \quad f(\Delta e) > z\_par \end{cases} \tag{3.36}$$



z_par: Schrankenparameter

Die auf die beiden verknüpften Knoten wirkenden Kräfte $\mathbf{F_i}$ und $\mathbf{F_j}$ setzen sich zusammen aus dem berechneten Kraftbetrag F des Verbindungselementes und der definierten vektoriellen Wirkungsrichtung $\mathbf{r}$:

$$\mathbf{F_i} = -F \cdot \mathbf{r}, \qquad \mathbf{F_j} = F \cdot \mathbf{r} \tag{3.37}$$

Erweiterte Verbindungselemente

Für eine realistische Einbeziehung mechanischer Eigenschaften in die Simulation sind weitere Typen von Verbindungselementen notwendig, die nicht der Definition 3.12 entsprechen. Dies betrifft die Dynamik der globalen Bewegungen, insbesondere das Verhalten der Z-Knoten.

Definition 3.17:

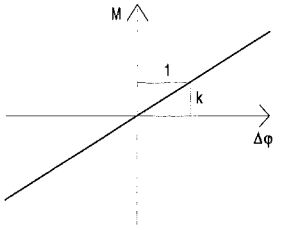
Erweiterte Verbindungselemente verknüpfen die Z-Vektoren eines oder mehrerer Masseknoten mit der charakteristischen Vektorfunktion f_C und liefern entsprechende Momentenvektoren/Kraftvektoren.

$$[\mathbf{M_i}, \mathbf{M_j}, \mathbf{M_k}, \ldots, \mathbf{F_i}, \mathbf{F_j}, \mathbf{F_k}, \ldots]^T = f_C(\mathbf{Z_i}, \mathbf{Z_j}, \mathbf{Z_k}, \ldots, par) \tag{3.38}$$

Beispiele für die Klasse der erweiterten Verbindungselemente sind das Torsionsfederelement oder das Biegefederelement. Diese werden insbesondere bei der Simulation partieller Deformationen und Verschiebungen sinnvoll eingesetzt (rohrförmige Geometrie, Kapitel 3.4).

Beispiel: Torsionsfederelement

$$\mathbf{M} = f(\Delta\phi)\cdot\mathbf{r} = k\cdot(\|\phi-\phi_0\|)\cdot\frac{\phi-\phi_0}{\|\phi-\phi_0\|} \tag{3.39}$$



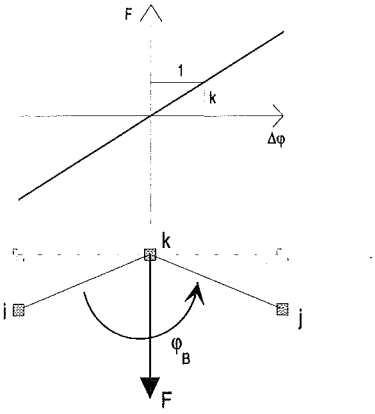
k: Proportionalitätsfaktor
$\mathbf{r}$: Drehrichtungsvektor
$\mathbf{M}$: Resultierender Drehmomentvektor
ϕ: Räumliche Orientierungsvektor des Knotens
ϕ_0: Ausgangsorientierung des Knotens, Referenzvektor

Beispiel: Biegefederelement

$$F = f(\Delta\varphi) = k\cdot(\varphi_B - \varphi_0) \tag{3.40}$$

$$\mathbf{F} = F\cdot\mathbf{r}, \qquad \mathbf{r} = (2\mathbf{u}_k - \mathbf{u}_i - \mathbf{u}_j)^\circ$$

$$\varphi_B = \angle((\mathbf{u}_k - \mathbf{u}_i),(\mathbf{u}_k - \mathbf{u}_j))$$



φ_B: Winkel zwischen den Richtungsvektoren vom Knoten k zu zwei Nachbarknoten i, j
φ_0: Ausgangswinkel des Biegefederelements
$\mathbf{u}_x$: Ortsvektor des Knotens x (x = i, j, k)
$\mathbf{r}$: Kraftrichtungsvektor
$\mathbf{F}$: Wirkende Kraft auf Knoten k

Die Spezifikation der Verbindungselemente innerhalb einer KISMET-Geometriedatei ist in Anhang A.3 erläutert. Bei der Implementierung wurden zwei Definitionsklassen berücksichtigt (Anhang B.2/B.3):

- lineare/quadratische Federelemente mit gleichen Eigenschaften
- allgemeine VE mit spezifisch definierten Eigenschaften

Mit dieser Klassifizierung verringert sich der Definitionsaufwand des deformierbaren Objekts, da Charakteristiken und Parameter für eine große Menge von Verbindungselementen gleich angenommen werden. Ähnlich wird auch bei der Definition der G-Knoten vorgegangen.

Die bisher aufgeführten Verbindungselemente sind passive Elemente, ihre resultierende Kraft ist allein von der Bewegung der verknüpften Knoten abhängig. Mit der MESD-Methodik sind jedoch auch aktive Elemente denkbar, die nach bestimmten Vorschriften und Funktionen eine interne Kraft auf die angekoppelten Masseknoten ausüben. Ein Anwendungsbeispiel solcher Aktivelemente ist die Simulation von Muskelbewegungen oder das Pulsieren von Gefäßen.

3.2.3 Topologie und Knotenbeziehungen

Definition 3.18:

Unter der **Topologie** im Sinne des MESD-Verfahrens versteht man die räumliche Lage und Anordnung der Knoten sowie die Art der Verknüpfung durch die Verbindungselemente. Hieraus ergibt sich die Struktur des nodalen Netzes, das den deformierbaren Körper repräsentiert.

Insbesondere ist auch die Verteilung der Masseknoten innerhalb des Objektes ein Aspekt der guten Approximation in topologischer Hinsicht. Hierbei stellt sich wieder die Frage, die schon im Kapitel 3.2.1 angesprochen wurde: wie minimiert man die Anzahl der Knoten, ohne ein unrealistisches Bewegungs- und Deformationsverhalten in Kauf nehmen zu müssen ?

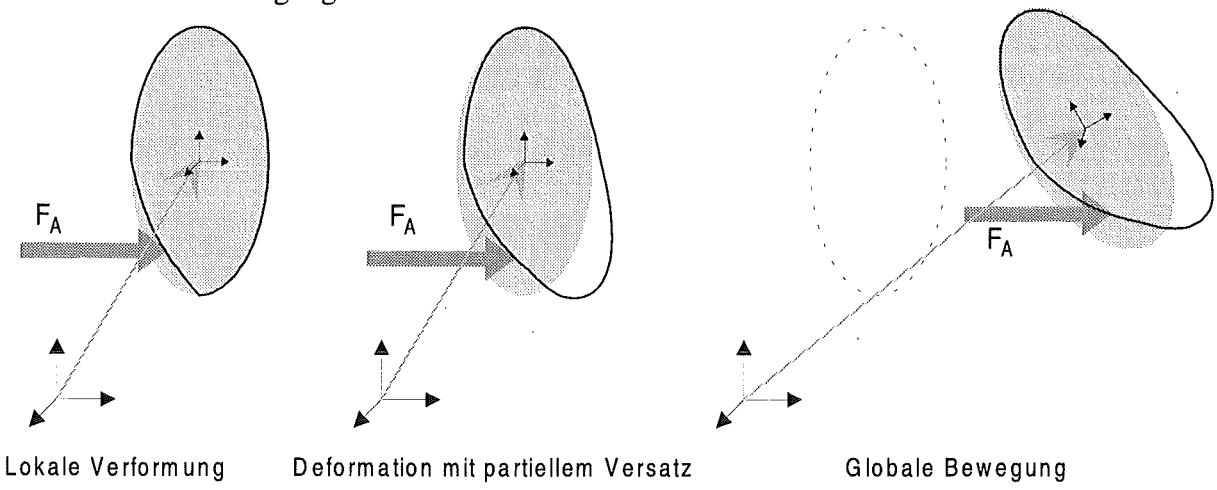


Bild 3.11: Klassen von Objektbewegungen

Das MESD-Verfahren nutzt einen mehrstufigen Ansatz, um diese Anforderung zu erfüllen. Die Basis bildet ein äußeres Netz, das die Oberfläche des Objektes repräsentiert und mit G-Knoten modelliert wird. Eine Deformation des Körpers äußert sich zunächst in einer lokalen Verformung der Oberfläche, dann in einer Verschiebung von Objektbereichen und schließlich in einer globalen Körperbewegung (Bild 3.11).

Diese drei Klassen von Bewegungen müssen bei der Modellbildung berücksichtigt werden. Betrachtet man die lokalen Verformungen unter dem Aspekt einer Energieänderung, so hat jede Verschiebung eines Oberflächenpunktes eine Erhöhung der potentiellen Energie des Körpers zur Folge. In einer kontinuierlichen Darstellung ergibt sich die Energiefunktion V:

$$V = \iint_{(A)} k_O \left(\int_{(x)} f_O(s,t,x)\,dx \right) ds dt \qquad (3.41)$$

s, t: Oberflächenvariablen
A: Körperoberfläche
x: Auslenkung des Körperpunktes von der Basislage, $x(s, t) = \|\mathbf{u} - \mathbf{u_0}\|$
k_O: Proportionalitätsparameter
f_O: Verformungsfunktion

Die Basislage $\mathbf{u_0}$ definiert die Ausgangslage des Punktes im Kräftegleichgewicht, also die Ruhelage der Punkte und damit des Systems. Zugrundegelegt wird hierbei stets ein objektfestes Bezugskoordinatensystem. Die wirkende Oberflächenkraft $\mathbf{F_O}$ resultiert aus dem Gradienten der Energiefunktion V (Prinzip der Minimierung der potentiellen Energie):

$$\mathbf{F_O} = -grad\ V \qquad (3.42)$$

Bei der Diskretisierung und Übertragung auf das nodale Netzmodell ergibt sich:

$$V = \sum_{i=1}^{n} V_i, \qquad V_i = \int_{(x_i)} f_O(i, x_i)\,dx_i \qquad (3.43)$$

V_i ist hierbei die Verformungsenergie für einen Masseknoten k_i. Nach Berücksichtigung von Gleichung (3.42) ergibt sich die Verformungskraft F_O, wobei die Kraftrichtung $\mathbf{r_i}$ vom Auslenkungsvektor des Knotens vorgegeben wird (Bild 3.12).

$$\mathbf{F_O} = -f_O(i, x_i) \cdot \mathbf{r_i}, \qquad \mathbf{r_i} = (\mathbf{u_i} - \mathbf{u_{i0}})^\circ \tag{3.44}$$

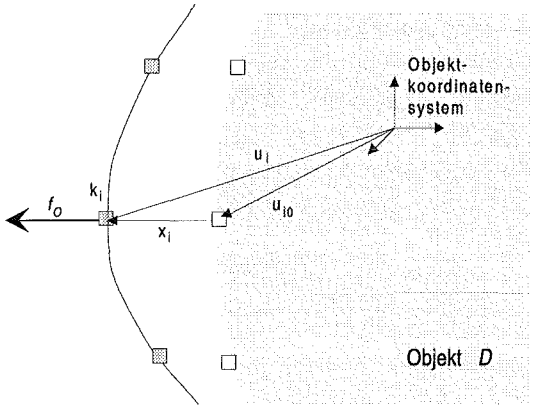


Bild 3.12: Verformungskraft eines Knotens

Die Verformungskraft wirkt zusätzlich zu den anderen Kräften auf einen Masseknoten, um das lokale volumenbedingte Verhalten des Objektes zu einzubeziehen. Durch diesen Ansatz wird eine große Anzahl innerer Knoten eingespart, außerdem wird der Stabilitätsbereich vergrößert. Generell ist für die Verformungsfunktion f_O eine beliebige Charakteristik möglich, äquivalent zu den Funktionen der Verbindungselemente. Aus Effizienz- und Stabilitätsgründen finden hier jedoch nur einfache lineare oder quadratische Funktionen Verwendung. Die Verformungskräfte und die Kräfte der Verbindungselemente überlagern sich zu einem dynamischen Gesamtsystem. Die Verformungsparameter sind hierbei Attribute der Masseknoten, analog zu der impliziten Dämpfung.

Knotenbeziehungen

Zur Realisierung partieller Deformationen von Objektregionen kann eine globale Verknüpfung der Oberflächenknoten mittels innerer Knoten und einer Vielzahl von Verbindungselementen durchgeführt werden. Wegen der großen Anzahl der nötigen Elemente ist dies jedoch nur für kleine Modelle praktikabel. Aus diesem Grunde verwendet das MESD-Verfahren einen Ansatz auf Basis von hierarchischen Knotenabhängigkeiten, der die Beziehungen zwischen Gruppen von Knoten festlegt.

Definition 3.19:

Eine Gruppe von Masseknoten (**Kindknoten**) kann als gemeinsamen Bezugsknoten einen besonders definierten **Vaterknoten** besitzen[4]. Dieser ist stets als Z-Knoten ausgeprägt. Der Vaterknoten wirkt auf seine Kindknoten positionsvorgebend.

Eine Bewegung des Vaterknotens besitzt also direkten Einfluß auf die aktuelle Position und die Basisposition des Kindknotens. Eine Bewegung findet stets relativ zum Vaterknoten statt, der somit der direkte Bezugspunkt der Menge der Kindknoten ist. Die Vaterknoten selbst können durch Verbindungselemente verknüpft werden. Außerdem können diese wiederum Kindknoten von einem Vaterknoten sein. Somit entsteht ein hierarchischer Beziehungsbaum,

[4] Die Relation Kindknoten-Vaterknoten ist keine Abbildung oder Funktion in mathematischem Sinne, da nicht jeder Knoten einen Vaterknoten besitzen muß. Jeder Kindknoten besitzt einen oder keinen Vaterknoten, ein Vaterknoten kann aber beliebig viele Kindknoten besitzen.

der theoretisch beliebig viele Schichten besitzen kann. Praktisch sinnvoll ist ein dreistufiger Baum, wobei der Wurzelknoten die globale Bewegung des deformierbaren Körpers repräsentiert (Objektknoten, Bild 3.13).

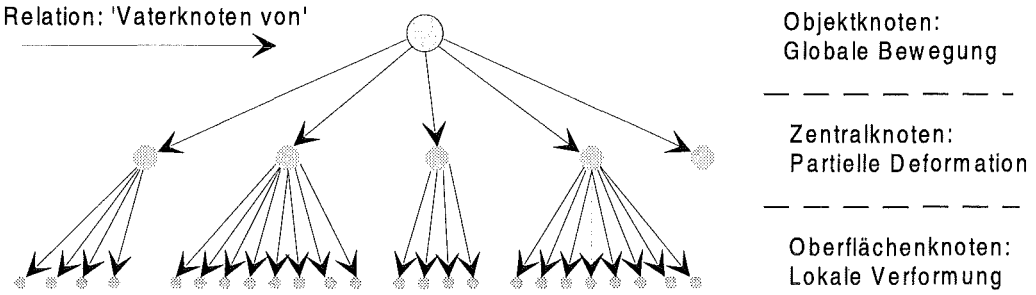


Bild 3.13: Beziehungsbaum der Knotengruppen

Als Eingangsgrößen der Vaterknoten werden neben den direkt verknüpften Verbindungselementen auch die Verformungskräfte der zugehörigen Kindknoten einbezogen, um partielle und globale Bewegungen zu übertragen. In Verbindung mit dem Abstandsvektor kann hieraus auch das auf den Vaterknoten wirkende Drehmoment abgeleitet werden (Bild 3.14).

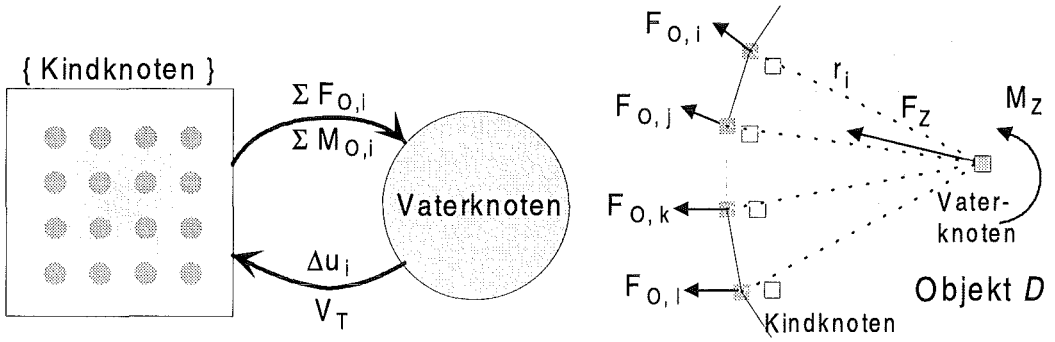


Bild 3.14: Strukturblock der Beziehung Vater-Kind-Knoten

Die Neuberechnung der Kindknoten-Positionen erfolgt in zwei Schritten mittels Koordinatentransformation über homogene Transformationsmatrizen [Kue91]:

1. Berechnung einer relativen Transformationsmatrix für den Vaterknoten

Diese (4x4)-Transformationsmatrix $\mathbf{V_T}$ beschreibt den Übergang von einem Koordinatensystem KS_j zu seinem Bezugskoordinatensystem KS_i. Möglich sind hierbei die Grundoperationen Translation (d_x, d_y, d_z) und Rotation (jeweils um die drei kartesischen Basisachsen: ϕ_x, ϕ_y, ϕ_z). Die relative Lage im Raum kann hiermit eindeutig spezifiziert werden. Als Bezug dient die Basislage des Vaterknotens (Bild 3.15).

$$\mathbf{V_T} = \begin{bmatrix} n_x & n_y & n_z & 0 \\ s_x & s_y & s_z & 0 \\ a_x & a_y & a_z & 0 \\ d_x & d_y & d_z & 1 \end{bmatrix} \tag{3.45}$$

n, s ,a: Kartesische Richtungsvektoren des Koordinatensystems KS_j in den Koordinaten des Systems KS_i, beinhalten die rotatorischen Grundoperationen

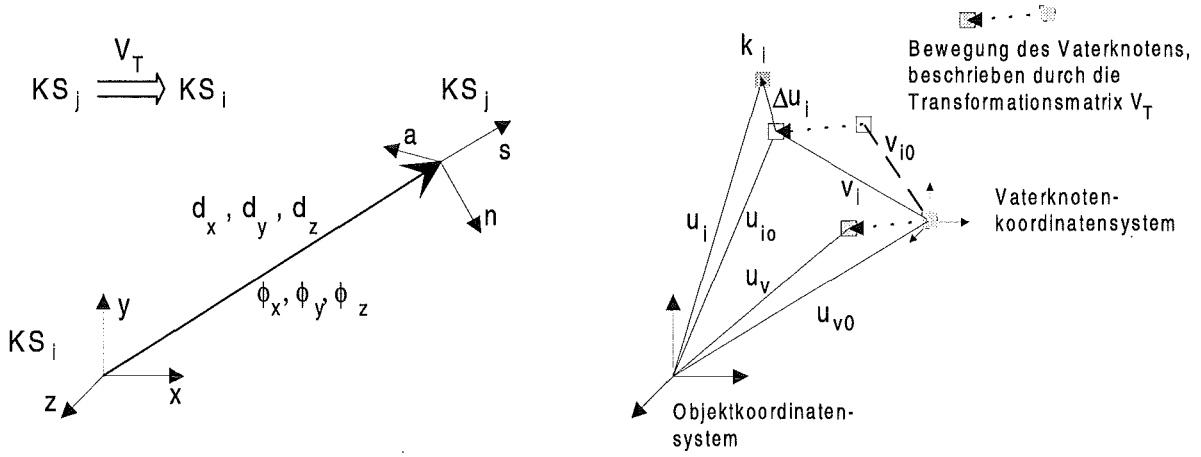


Bild 3.15: Bezugskoordinatensysteme und Koordinatentransformation

2. Anpassung der relativen Lage aller Kindknoten i=1..n durch Transformation der Positionsvektoren

$$\mathbf{v}_i = \mathbf{v}_{io}\mathbf{V}_T \tag{3.46}$$

$\mathbf{v}_i$: lokaler Verschiebungsvektor des Kindknotens in homogenen Koordinaten $\mathbf{v}_i = [v_{ix}, v_{iy}, v_{iz}, 1]^T$
$\mathbf{v}_{io}$: lokaler Basisvektor des Kindknotens (homogene Koordinaten)

Zu beachten ist, daß der Vektor $\mathbf{v}_{io}$ die Basisposition des Kindknoten relativ zum Vaterknoten angibt. Für die neue Basisposition $\mathbf{u}'_{i0}$ im Objektkoordinatensystem ist daher die relative Position des Vaterknotens $\mathbf{u}_{v0}$ zu berücksichtigen.

$$\mathbf{u}'_{io} = \mathbf{v}_i + \mathbf{u}_{v0} \tag{3.47}$$

Die Basislagen der Kindknoten müssen aktualisiert werden, hierdurch erfolgt eine Verschiebung des energetischen Minimums des Verformungspotentials. Der Vaterknoten dient sozusagen als 'Aufhängung'. Mit ihm bewegt sich eine 'Schale' von Kindknoten mit, die relativ zu ihrem Vaterknoten Bewegungen ausführen können. Der neue Positionsvektor $\mathbf{u}_i$' des Kindknotens wird somit bestimmt durch:

$$\mathbf{u}'_i = \mathbf{u}'_{i0} + \Delta\mathbf{u}_i, \qquad \Delta\mathbf{u}_i = \mathbf{u}_i - \mathbf{u}_{i0} \tag{3.48}$$

Die vorgestellte Methodik kann den Einsatz von inneren Knoten minimieren. Trotzdem ist es zur Erzielung eines realistischen Modellverhaltens in vielen Fällen notwendig, einzelne oder auch schichtweise innere Knoten einzufügen. Hierzu werden die F-Knoten verwendet.

Bemerkung 3.3:

Je höher die lokale Komplexität innerhalb des Körpers ist, desto genauer soll die Modellierung und damit auch die Anzahl der Knoten in dieser Region sein. Ein Maß hierfür kann der Betrag des Oberflächen-Gradienten sein.

Für die Modellbildung deformierbarer Objekte bieten sich regelmäßige Netze an, die aufgrund der gleichförmigen Struktur die Modellierung vereinfachen. Die Knoten werden mit Hilfe einer Matrix organisiert, wobei jedes Matrixelement einen G-Knoten repräsentiert. Die Netzmaschen können in Form von Vierecken oder Dreiecken ausgeprägt sein, wobei die Maschengröße den lokalen Anforderungen angepaßt werden. Durch diese Netzstruktur wird außerdem die computergraphische Darstellung als Freiformfläche wesentlich erleichtert

(Kapitel 5). Bei der Netzbildung sind unter Verwendung von Diagonal-Verbindungselementen verschiedene topologische Grundformen möglich (Bild 3.16):

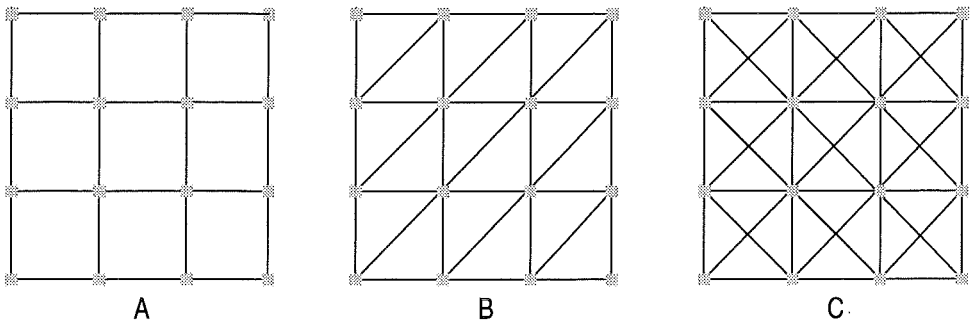


Bild 3.16: Topologievarianten des Oberflächennetzes

Die Version C hat sich im praktischen Einsatz als beste Lösung herausgestellt, da hierbei keine unplausiblen Scherungseffekte auftreten. Weiterhin wird das Oberflächenverhalten besser nachgebildet, da alle Punkte innerhalb eines Netzvierecks durch VE verbunden sind.

3.2.4 Globale Verknüpfung und globaldynamische Objekte

In den bisherigen Ausführungen wurden einzelne, bezüglich der Umgebung unabhängige Körper betrachtet. In einem Simulationsszenario sind im allgemeinen jedoch mehrere deformierbare Objekte vorhanden. Diese können in gegenseitiger Wechselwirkung stehen, sich also dynamisch beeinflussen. Weiterhin ist es vorteilhaft, komplexe deformierbare Strukturen in getrennte, einfacher zu beschreibende Teilsysteme aufzuteilen und diese dann zu koppeln.

Für jedes deformierbare Objekt existiert ein lokales Bezugskoordinatensystem, auf das sich alle in der Definitionsmenge enthaltenen Knoten beziehen. Hierdurch wird jedoch eine globale Verknüpfung verschiedener Objekte erschwert. Notwendig ist daher die Einführung eines globalen Bezugskoordinatensystems für das gesamte Simulationsszenario. Mit Hilfe von Transformationsmatrizen $\mathbf{T}_i$ lassen sich die Objektkoordinatensysteme KS_i in das globale Zentralsystem KS_0 überführen. Zur Kopplung verschiedener Objekte müssen alle Knotenvektoren und Kraftvektoren in das globale System transformiert, miteinander verknüpft und in das jeweilige Bezugssystem zurückgeführt werden.

Für die dynamische Kopplung existieren innerhalb des MESD-Verfahrens zwei verschiedene Ansätze: die Verknüpfung über gemeinsame ('globale') Verbindungselemente oder die Verwendung gemeinsamer Knoten.

Globale Verbindungselemente

Äquivalent zur Verknüpfung zweier Knoten innerhalb eines Körpers lassen sich zwei Knoten auch objektübergreifend durch ein Verbindungselement verknüpfen. Analog zu Definition 3.12 läßt sich daher bestimmen:

Definition 3.20:

Ein **globales Verbindungselement** (GVE) verknüpft einen Masseknoten k_i des Objektes D_m mit einem Masseknoten k_j des Objektes D_n. Das globale Verbindungselement übt dabei auf beide Knoten eine Kraft aus, die Eigenschaften des GVE entsprechen den in Kapitel 3.2.2 eingeführten Charakteristiken für lokale VE.

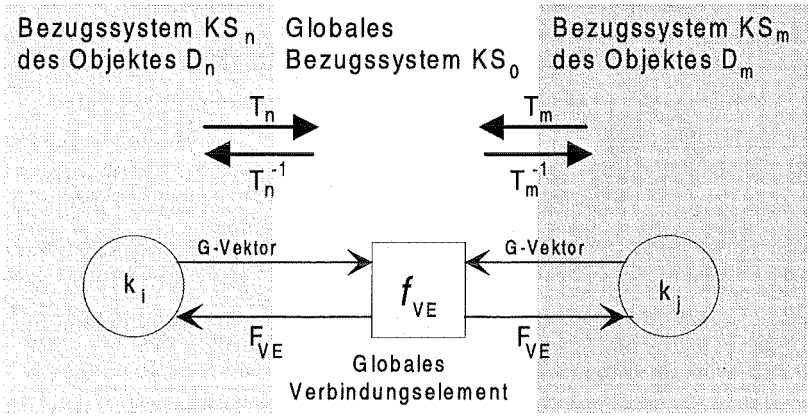


Bild 3.17: Prinzip der Verknüpfung über ein globales Verbindungselement

Bild 3.17 erläutert die prinzipielle Vorgehensweise der Transformation und Verknüpfung. Für die Masseknoten ergibt sich somit eine weitere Kraftkomponente, die sich den lokalen VE-Kräften überlagert. Als Sonderfall ist es möglich, daß ein Knoten kein Bestandteil eines Objektes ist, sondern ein fester Punkt im dreidimensionalen Modellraum. Hierdurch sind globale Bindungen und Bewegungsbeschränkungen möglich.

Gemeinsame Knoten

Eine unmittelbare Kopplung zweier Objekte ist über gemeinsame Knoten möglich. Dies bewirkt eine punktweise starre Verbindung. Aus Implementierungsaspekten (Datenstruktur, Kapitel 3.4) ist es jedoch nicht effizient, daß zwei Objekte auf einen Knoten verweisen und diesen in die jeweilige lokale Netzdynamik einbinden. Das MESD-Verfahren geht daher zunächst von einem Knotenpaar aus, dessen Einzelknoten in ihrem jeweiligen Netzsystem integriert sind. In einem zusätzlichen Schritt werden diese beiden 'Hyperknoten' miteinander verkettet, so daß sie durch einen gemeinsamen G-Vektor beschrieben werden können und damit die gleiche Kinematik besitzen. Zur Vereinfachung der Berechnung wird von zwei differenzierten Knotenklassen ausgegangen.

Definition 3.21:

Zwei deformierbare Objekte können über sogenannte **Hyper-Knoten** dynamisch zusammengefügt werden. Der **Master-Knoten** ist der positionsvorgebende Knoten der Kopplung. Sein G-Vektor dient zur Beschreibung der gemeinsamen Kinematik. Der **Slave-Knoten** ist der kraftrückgebende Knoten der Beziehung. Sein resultierender Kraftvektor wird dem Master-Knoten als zusätzliche Eingangsgröße beaufschlagt.

Bei der Simulation stellt sich zwischen den Hyperknoten ein dynamisches Gleichgewicht ein. Die verschiedenen Objektkoordinatensysteme müssen bei der Verknüpfung berücksichtigt werden (Bild 3.18). Anstelle eines beweglichen Knotens kann auch ein fester Punkt im Raum als positionsvorgebender Master dienen, so daß der Slave-Knoten ortsfest im Simulationsszenario verankert ist.

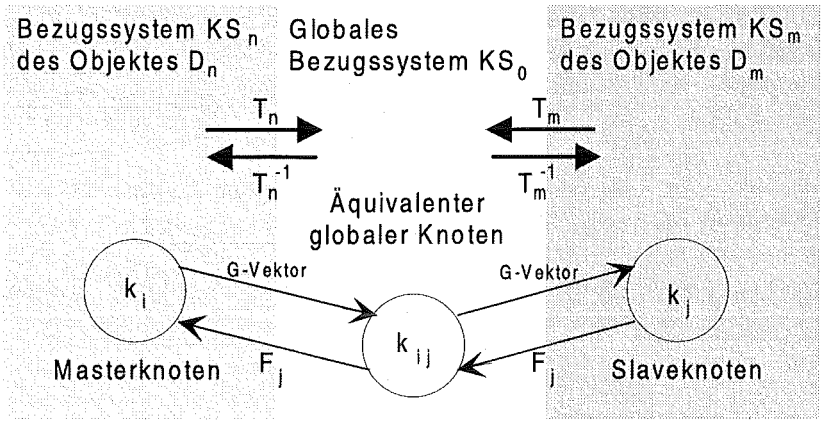


Bild 3.18: Prinzip der Verknüpfung über ein gemeinsames Knotenpaar

Globaldynamische Objekte

Die Modellkomplexität in einem Simulationsszenario unterliegt den Grenzen der Echtzeitfähigkeit, die vom verwendeten Rechnersystem vorgegeben sind. Im Gesamtszenario sollte daher eine spezifische Abstufung der Modellierungsdetails erfolgen, wobei die Bedingungen durch die Zielsetzung der Simulation festgelegt werden. Dies führt dazu, daß Objekte mit stark vereinfachtem physikalischen Verhalten modelliert werden können, wenn es die Anforderungen an die Simulation erlauben. Ein Beispiel hierfür ist die Integration von Objekten als approximierte Starrkörper in das globale Szenario.

Definition 3.22:

Ein **globaldynamisches Objekt** ist ein dreidimensionaler, geometrischer Körper mit spezifiziertem globalem dynamischen Verhalten. In sich ist der Körper nicht deformierbar, sondern behält seine starre Form. Die Bewegungen können somit durch die Starrkörperdynamik beschrieben werden.

Repräsentiert wird ein globaldynamisches Objekt durch einen Z-Knoten, der beliebig innerhalb des Körpers lokalisiert sein kann. Attribute sind die Masse, die Hauptträgheitsmomente sowie die implizite Dämpfungskonstante des Knotens. Die Objektbewegung wird vollständig durch den Z-Vektor beschrieben. Weitere ortsfeste Knoten können frei bestimmt werden, durch die Beschränkung aller Freiheitsgrade sind diese eindeutig durch den Positionsvektor relativ zum Z-Knoten spezifiziert. An alle Knoten sind globale Verbindungselemente anknüpfbar, die eine dynamische Kopplung zu anderen (deformierbaren oder globaldynamischen) Objekten realisieren. Auch eine Bindung über Hyperknoten ist möglich, wobei allerdings der Slave-Knoten stets zu einem deformierbaren Objekt gehören muß.

Innerhalb von KISMET ist für den repräsentierten Körper jede beliebige Art der Geometriedefinition möglich [Kue91], da ein globaldynamisches Objekt durch funktionelle Erweiterung des Objektkoordinatensystems ('Frame') bestimmt wird. Die Definition von globalen Verknüpfungen und globaldynamischer Objekte innerhalb KISMET wird in einer speziellen Datei ('ELADYN_HYPER') abgelegt, welche durch ein Skript-Kommando festgelegt wird (Anhang C.1). Die Spezifikation ist in Anhang B.4 ausführlich erläutert.

3.3 Numerik der Simulation

Ausgehend von der physikalischen Beschreibung deformierbarer Objekte wurde im letzten Kapitel die Modellbildung auf Basis des MESD-Verfahrens hergeleitet und erläutert. Im Hinblick auf die Simulation auf einem Rechnersystem ist eine Modellaufbereitung notwendig, damit die Verfahren in Algorithmen umgesetzt werden können. Dies impliziert die Erstellung eines mathematischen Modells, welches den Sachverhalt formal beschreibt und effiziente numerische Lösungsverfahren ermöglicht.

3.3.1 Mathematisches Modell

Ziel der Simulation ist die Berechnung der Bewegungsgleichungen für alle Knoten des Objektmodells. Aufgrund der Systemkomplexität und Interaktivität sind sicherlich keine analytischen Lösungen in Form von geschlossenen Gleichungen möglich. Daher sind numerische Verfahren anzuwenden, die die Bewegung an zeitlich diskreten Stellen auswerten und die Knotenpositionen bestimmen.

Gegeben ist ein deformierbares Objekt $D = \{K, V\}$. Die Mächtigkeit der Knotenmenge K ist $n = |K|$. Ausgehend von Gleichung (3.17) kann das dynamische Gesamtsystem beschrieben werden durch eine allgemeines System von Differentialgleichungen 2. Ordnung:

$$\mathbf{M}\ddot{\mathbf{x}} + \boldsymbol{D}_{\mathrm{K}}(\mathbf{x}, \dot{\mathbf{x}}) + \boldsymbol{F}(\mathbf{x}, \dot{\mathbf{x}}) = \mathbf{F}_{\mathrm{A}} \tag{3.49}$$

Hierbei repräsentiert der Funktionenvektor $\boldsymbol{D}_{\mathrm{K}}$ die Dämpfung sowie $\boldsymbol{F}$ die inneren wirkenden Kräfte des Systems. $\boldsymbol{M}$ ist die Massenmatrix, die bei nodalen Netzen stets als reine Diagonalmatrix ausgeprägt ist. Sie setzt sich aus den Teilmatrizen $\boldsymbol{M}_{\mathrm{i}}$ (i=1...n) zusammen, $\boldsymbol{M}_{\mathrm{i}}$ symbolisiert die Matrixdarstellung der Masse m_i des Knotens i.

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{M}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{M}_2 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{M}_3 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{M}_{\mathrm{n}} \end{bmatrix} \qquad \boldsymbol{M}_{\mathrm{i}} = \begin{bmatrix} m_i & 0 & 0 \\ 0 & m_i & 0 \\ 0 & 0 & m_i \end{bmatrix} \tag{3.50}$$

Die von außen auf das System einwirkenden Kräfte werden durch $\mathbf{F}_{\mathrm{A}}$ erfaßt. Der Systemvektor $\mathbf{x}$ setzt sich aus den dreidimensionalen Positionsvektoren aller Knoten zusammen, er besitzt somit die Dimension (3n).

$$\mathbf{x} = \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_{\mathrm{n}} \end{bmatrix}, \qquad \mathbf{u}_{\mathrm{i}} = \begin{bmatrix} \mathrm{u}_{\mathrm{ix}} \\ \mathrm{u}_{\mathrm{iy}} \\ \mathrm{u}_{\mathrm{iz}} \end{bmatrix} \tag{3.51}$$

Wie in Kapitel 3.2 eingeführt, wird innerhalb des MESD-Verfahrens jeder Masseknoten implizit gedämpft, wobei eine lineare Proportionalität zur Knotengeschwindigkeit vorausgesetzt wird (viskose Dämpfung, Parameter d_i). Die Dämpfungsfunktion $\boldsymbol{D}_{\mathrm{K}}$ vereinfacht sich somit zu

$$\boldsymbol{D}_K(\mathbf{x}, \dot{\mathbf{x}}) = \boldsymbol{D} \cdot \dot{\mathbf{x}} \tag{3.52}$$

wobei die Dämpfungsmatrix $\boldsymbol{D}$ wiederum nur Diagonalelemente besitzt. Äquivalent zur Massenmatrix lassen sich wieder Untermatrizen $\boldsymbol{D}_i$ definieren, welche die viskose Dämpfung der Knoten charakterisieren:

$$\boldsymbol{D} = \begin{bmatrix} \boldsymbol{D}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{D}_2 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{D}_3 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{D}_n \end{bmatrix} \qquad \boldsymbol{D}_i = \begin{bmatrix} d_i & 0 & 0 \\ 0 & d_i & 0 \\ 0 & 0 & d_i \end{bmatrix} \tag{3.53}$$

Definition 3.23:

Das kinematische und elastodynamische Verhalten eines deformierbaren Körpers D läßt sich auf Basis des MESD-Verfahrens mit einem nichtlinearen Differentialgleichungssystem 2. Ordnung beschreiben:

$$\boldsymbol{M}\ddot{\mathbf{x}} + \boldsymbol{D}\dot{\mathbf{x}} + \boldsymbol{F}(\mathbf{x}, \dot{\mathbf{x}}) = \mathbf{F}_A \tag{3.54}$$

Die Eigenschaften der Verbindungselemente sind in dem Kraftfunktionsvektor $\mathbf{F}$ enthalten. Die Systemmatrizen $\boldsymbol{M}$ und $\boldsymbol{D}$ sowie der Systemvektor $\mathbf{x}$ repräsentieren die Eigenschaften und Zustände der Masseknoten.

Der innere Kraftfunktionsvektor $\boldsymbol{F}$ repräsentiert die Gesamtheit der Verbindungselemente V. Aufgrund der Vielfalt der möglichen VE wird dieser Kraftvektor im allgemeinen sehr komplex. Das Differentialgleichungssystem ist daher nichtlinear und stark gekoppelt, was die Lösung unter Echtzeitanforderungen erheblich erschwert.

An dieser Stelle muß erwähnt werden, daß auch die Rotation der Z-Knoten in die beschreibenden Gleichungen einfließt. Die mathematische Struktur entspricht jedoch der des obigen Translationssystems, so daß die Lösungsverfahren äquivalent sind und allgemein übertragen werden können[5].

$$\boldsymbol{J}_R\ddot{\theta} + \boldsymbol{D}_R\dot{\theta} + \boldsymbol{M}_R(\theta, \dot{\theta}, \mathbf{x}, \dot{\mathbf{x}}) = \mathbf{M}_A \tag{3.55}$$

θ: Systemrotationsvektor, beinhaltet die Einzelvektoren ϕ_i
$\boldsymbol{J}_R$: Trägheitmomentenmatrix
$\boldsymbol{D}_R$: Rotationsdämpfungsmatrix
$\boldsymbol{M}_R$: Momentenfunktionsvektor
$\mathbf{M}_A$: Externer Momentenvektor

Eine implizite Lösung des Systems (3.54) bedarf iterativer Verfahren, da aufgrund der Nichtlinearitäten des Kraftfunktionsvektors direkte Methoden nicht einsetzbar sind [BS87]. Iterationsverfahren sind jedoch für Echtzeitanwendungen ungeeignet, da keine beschränkten Rechenzeiten garantiert werden können, zumindest bei vorgegebenen festen Fehlerschranken. Bei großen Systemen ist die numerische Konvergenz nur bei entsprechend hohen Rechenzeiten gewährleistet.

Zur Sicherung der Echtzeitfähigkeit muß eine implizite Lösung umgangen werden. Hierzu ist es notwendig, das System von Differentialgleichungen (3.54) in diskrete Differentialgleichungen (DGL) aufzuteilen, also einzelne Knoten i zu betrachten. Die dem Knoten

[5] Eine Verallgemeinerung des beschreibenden Differentialgleichungssystems auf translatorische und rotatorische Bewegungen ist durch entsprechende Erweiterung der Vektoren und Matrizen möglich.

entsprechenden Zeilen des inneren Kraftfunktionsvektors $\boldsymbol{F}$ gehen dabei in einen äußeren Knotenkraftvektor f über:

$$m_i\ddot{\mathbf{u}}_i + d_i\dot{\mathbf{u}}_i = \mathbf{F}_A + f(\mathbf{x},\dot{\mathbf{x}}) \tag{3.56}$$

Die DGL (3.56) ist jedoch nicht entkoppelt, da die Knotenkraftfunktion entsprechend der Verknüpfungen über Verbindungselemente noch den Systemvektor $\mathbf{x}$ als Argument enthält. Der Vektor $\mathbf{x}$ wiederum enthält auch den Knotenvektor $\mathbf{u_i}$, so daß es sich immer noch um eine implizite Gleichung handelt.

3.3.2 Numerische Lösungsverfahren

Eine Differentialgleichung 2. Ordnung kann in zwei Differentialgleichungen 1. Ordnung aufgeteilt werden. $\mathbf{v_i}$ ist hierbei der Geschwindigkeitsvektor des Knotens i:

$$\dot{\mathbf{u}}_i = \frac{d\mathbf{u}_i}{dt} = \mathbf{v}_i \tag{3.57}$$

$$m_i\dot{\mathbf{v}}_i + d_i\mathbf{v}_i = \mathbf{F}_A + f(\mathbf{x},\dot{\mathbf{x}}) \tag{3.58}$$

Die DGL (3.57) ist eine einfache lineare Differentialgleichung, die für sich alleine genommen entkoppelt und damit durch bekannte numerische Integrationsverfahren lösbar ist. Zur weiteren Behandlung der DGL (3.58) muß an dieser Stelle der Übergang in eine zeitdiskrete Darstellung erfolgen. Zur besseren Lesbarkeit und Vereinfachung der Schreibweise wird der Index i weggelassen, welcher den Knoten i innerhalb des nodalen Systems spezifiziert. Hierfür werden die diskreten Zeitindizes $T \subset \mathbb{N} \cup \{0\}$ eingeführt, wobei t_k den aktuellen absoluten Zeitpunkt darstellt. Der Zeitschritt der numerischen Integration wird mit h bezeichnet, er wird in dieser Herleitung als konstant angenommen[6].

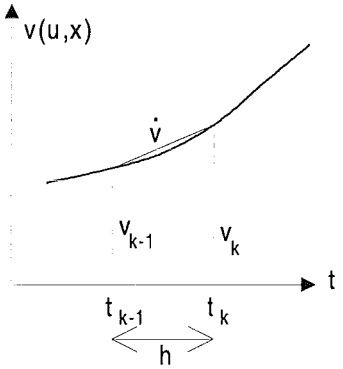


Bild 3.19: Sekantenverfahren

$$h = t_k - t_{k-1} = t_{k+1} - t_k, \qquad k \in T \tag{3.59}$$

$$m\dot{\mathbf{v}}_k + d\mathbf{v}_k = \mathbf{F}_{A,k} + f_k(\mathbf{x}_k,\dot{\mathbf{x}}_k) \tag{3.60}$$

Der Rückwärtsdifferenzenquotient (3.61) wird aus der Sekantenregel der numerischen Differentiation abgeleitet (Bild 3.19), wobei die Ableitung dv/dt durch ein Geradenstück approximiert wird. Dieser einfache Quotient stellt nur bei kleiner Schrittweite h eine gute Näherung dar.

$$\dot{\mathbf{v}}_k = \frac{\mathbf{v}_k - \mathbf{v}_{k-1}}{h} \tag{3.61}$$

Einsetzen von (3.61) in (3.60) ergibt die Differenzengleichung:

$$\frac{m}{h}(\mathbf{v}_k - \mathbf{v}_{k-1}) + d\mathbf{v}_k = \mathbf{F}_{A,k} + f_k(\mathbf{x}_k,\dot{\mathbf{x}}_k) \tag{3.62}$$

und daraus die Berechnungsvorschrift für den neuen Geschwindigkeitsvektor:

$$\mathbf{v}_k = \frac{1}{1+\frac{d\cdot h}{m}}\left(\mathbf{v}_{k-1} + \frac{h}{m}(\mathbf{F}_{A,k} + f_k(\mathbf{x}_k,\dot{\mathbf{x}}_k))\right) \tag{3.63}$$

[6] Die Schrittweite h wird in der Echtzeitsimulation zur Abtastrate h_i, die von der Rechenleistung und dem Simulationsablauf abhängig und daher im allgemeinen nicht konstant ist. In Kapitel 3.4 wird darauf eingegangen.

Analog läßt sich aus (3.57) der Positionsvektor $\mathbf{u}_k$ berechnen:

$$\mathbf{u}_k = \mathbf{u}_{k-1} + h \cdot \mathbf{v}_k \tag{3.64}$$

Genau betrachtet ist (3.63) immer noch eine implizite Gleichung, da der Systemvektor $\mathbf{x}_k$ als Argument des Kraftfunktionsvektors f_k immer noch den Geschwindigkeitsvektor $\mathbf{v}_k$ und den daraus zu berechnenden Positionsvektor $\mathbf{u}_k$ enthält. Setzt man voraus, daß die Schrittweite h sehr klein und/oder das Gesamtsystem stark gedämpft ist ($d >> m/h$), so läßt sich zur Kraftberechnung der einen Zeitschritt zuvor berechnete Systemvektor $\mathbf{x}_k$ verwenden[7]. Diese Voraussetzung sollte bei einem genügend leistungsstarken Rechnersystem stets erfüllt sein.

$$f_k(\mathbf{x}_k, \dot{\mathbf{x}}_k) \cong f_k(\mathbf{x}_{k-1}, \dot{\mathbf{x}}_{k-1}) \tag{3.65}$$

Unter dieser Annahme wird Gleichung (3.63) zeitlich entkoppelt, der Lösungsvektor $\mathbf{v}_k$ kann explizit bestimmt werden. Hierbei handelt es sich um ein Anfangswertproblem, die Startvektoren $\mathbf{u}_0$ und $\mathbf{v}_0$ müssen vorgegeben werden. Bei der Simulation wird stets von einer stabilen Ruhelage ausgegangen ($\mathbf{v}_0 = \mathbf{0}$, $\mathbf{u}_0 = \mathbf{u}(\Sigma F_A = 0)$).

Durch Einführung einer Beschränkungsmatrix können einzelne Freiheitsgrade im dreidimensionalen Bezugssystem beschränkt werden, um die lokale Bewegung von Masseknoten zu begrenzen. Die Beschränkungsmatrix mit ihren drei Komponenten b_x, b_y, b_z ist ein weiteres Attribut der Masseknoten.

$$\boldsymbol{B} = \begin{bmatrix} b_x & 0 & 0 \\ 0 & b_y & 0 \\ 0 & 0 & b_z \end{bmatrix} \qquad b_x, b_y, b_z \in [0,1] \tag{3.66}$$

Zusammengefaßt läßt sich somit festhalten:

Definition 3.24:

Innerhalb des MESD-Verfahrens lassen sich der Positionsvektor $\mathbf{u}$ und der Geschwindigkeitsvektor $\mathbf{v}$ eines Knotens des deformierbaren Objektes D zum Zeitpunkt t_k über die folgenden sukzessiven Berechnungsvorschriften bestimmen:

$$\mathbf{v}_k = \frac{1}{1 + \frac{d \cdot h}{m}} \left(\mathbf{v}_{k-1} + \frac{h}{m}(\mathbf{F}_{A,k} + f_k(\mathbf{x}_{k-1}, \dot{\mathbf{x}}_{k-1}))\right) \tag{3.67}$$

$$\mathbf{u}_k = \mathbf{u}_{k-1} + \boldsymbol{B} \cdot h \cdot \mathbf{v}_k \tag{3.68}$$

Zur vollständigen kinematischen Beschreibung des Objektes muß der Algorithmus innerhalb eines Zeitschrittes für jeden Knoten der Knotenmenge K durchlaufen werden.

Der vorgestellte Lösungsalgorithmus ist speziell für gedämpfte nodale Systeme geeignet, es wird eine kleine Zeitschrittweite erwartet (Größenordnung h ≈ 10 ms). Die Komplexität des Verfahrens ist proportional zu O(n), so daß es insbesondere für die Echtzeitsimulation sehr gut geeignet ist (n: Mächtigkeit der Knotenmenge K). Durch die Vereinfachungen ist die Genauigkeit des Algorithmus eingeschränkt, ohne jedoch der Anforderungsspezifikation zu widersprechen. In Kapitel 3.3.3 werden weitere Lösungsverfahren untersucht und eine vergleichende Bewertung durchgeführt.

[7] Wie weiter hinten gezeigt wird, wird diese Näherung auch bei den gängigen Lösungsverfahren für Anfangswertprobleme implizit vorausgesetzt.

Die auf der rechten Seite von Gleichung (3.67) auftretenden Kräfte setzen sich zusammen aus einer von außen eingeprägten Kraft $\mathbf{F}_A$ und dem Knotenkraftvektor f. Dieser setzt sich aus den im vorigen Kapitel eingeführten Komponenten zusammen:

$$f(\mathbf{x},\dot{\mathbf{x}}) = \Sigma f_{VE}(\mathbf{x},\dot{\mathbf{x}}) + \Sigma f_{KK}(\mathbf{x},\dot{\mathbf{x}}) + f_O(\mathbf{u},\dot{\mathbf{u}}) + \mathbf{T}^{-1}(\Sigma f_{GVE}(\mathbf{x},\dot{\mathbf{x}}) + \Sigma f_{Hyp}(\mathbf{x},\dot{\mathbf{x}}) + \mathbf{F}_G) \quad (3.69)$$

Σf_{VE}: Resultierende Kraft der den Knoten verknüpfenden Verbindungselemente (=: $\mathbf{F}_{VE}$)
Σf_{KK}: bei Vaterknoten: Resultierende Kraft aller Verformungskräfte von Kindknoten (=: $\mathbf{F}_{KK}$)
f_O: Verformungskraft des Knotens
$\mathbf{T}^{-1}$: Objekttransformationsmatrix: Globales Koordinatensystem → Objektkoordinatensystem
Σf_{GVE}: Resultierende Kraft der globalen Verbindungselemente des Knotens (=: $\mathbf{F}_{GVE}$)
Σf_{Hyp}: Resultierende Kraft der verknüpften Hyperknoten: Slave-Kraft (=: $\mathbf{F}_{Hyp}$)
$\mathbf{F}_G$: Gravitationskraft, $\mathbf{F}_G = m \cdot \mathbf{g}$

Die Gravitationskraft ist optional je nach Szenario zu aktivieren (Attribut des Objektes). Die Erdbeschleunigung $\mathbf{g}$ zeigt stets in Richtung der negativen y-Achse des globalen Weltkoordinatensystems. Hinzu kommen noch eventuell auftretende Kollisionskräfte, die in Kapitel 4 behandelt werden.

Zum Vergleich soll an dieser Stelle kurz das klassische Euler-Cauchy-Verfahren zur Lösung von Differentialgleichungen der Form $dy/dt = g(y,t)$ hergeleitet werden. Ausgegangen wird hierbei von der Taylor-Entwicklung der Funktion y [Sch88]:

$$y(t+h) = y(t) + h \cdot \dot{y}(t) + \frac{1}{2}h^2 \cdot \ddot{y}(t) + \frac{1}{6}h^3 \cdot y^{(3)}(t) + O(h^4) \quad (3.70)$$

Setzt man voraus, daß die Schrittweite h sehr klein ist ($h << 1$), so lassen sich die Glieder mit den Faktoren h^i für die Potenzen $i \geq 2$ vernachlässigen. Man kann also vereinfacht schreiben:

$$y(t+h) \equiv y(t) + h \cdot \dot{y}(t) \quad (3.71)$$

Überträgt man Gleichung (3.60), so ergeben sich die folgenden Übergänge:

$$\begin{aligned} &y(t+h) \rightarrow \mathbf{v}_k, \qquad y(t) \rightarrow \mathbf{v}_{k-1}, \\ &\dot{y}(t) = g(y,t) \rightarrow \frac{1}{m}(\mathbf{F}_{A,k-1} + f_{k-1}(\mathbf{x}_{k-1},\dot{\mathbf{x}}_{k-1}) - d \cdot \mathbf{v}_{k-1}) \end{aligned} \quad (3.72)$$

Hiermit erhält man die Lösungsgleichung für die Knotengeschwindigkeit auf Basis des Verfahrens von Euler-Cauchy:

$$\mathbf{v}_k = (1 - \frac{d \cdot h}{m}) \cdot \mathbf{v}_{k-1} + \frac{h}{m}(\mathbf{F}_{A,k-1} + f_{k-1}(\mathbf{x}_{k-1},\dot{\mathbf{x}}_{k-1})) \quad (3.73)$$

Nach Umformung der Parameter ergibt sich:

$$\mathbf{v}_k = (1 - \frac{d \cdot h}{m}) \cdot (\mathbf{v}_{k-1} + \frac{h}{m}(\frac{1}{1 - \frac{d \cdot h}{m}})(\mathbf{F}_{A,k-1} + f_{k-1}(\mathbf{x}_{k-1},\dot{\mathbf{x}}_{k-1}))) \quad (3.74)$$

Für sehr kleine Faktoren $\frac{d \cdot h}{m} << 1$ gehen Gleichungen (3.67) und (3.74) ineinander über. Der Unterschied beider Lösungsverfahren liegt im wesentlichen in der Berücksichtigung der viskosen Dämpfung. Während das MESD-Verfahren im Lösungsalgorithmus den aktuellen Dämpfungsterm implizit verwendet (3.62), berücksichtigt das klassische Euler-Verfahren den Term nur als Funktion der zuletzt berechneten Geschwindigkeit (3.72). Daher ist beim modifizierten Euler-Verfahren (3.67) eine stabileres Verhalten zu erwarten als bei dem klassischen Euler-Cauchy-Verfahren (3.74). Dies wird durch einen Vergleich im nächsten

Kapitel belegt. Bei der Lösungsgleichung (3.66) zur Berechnung des Knotenpositionsvektors entsprechen sich beide Verfahren.

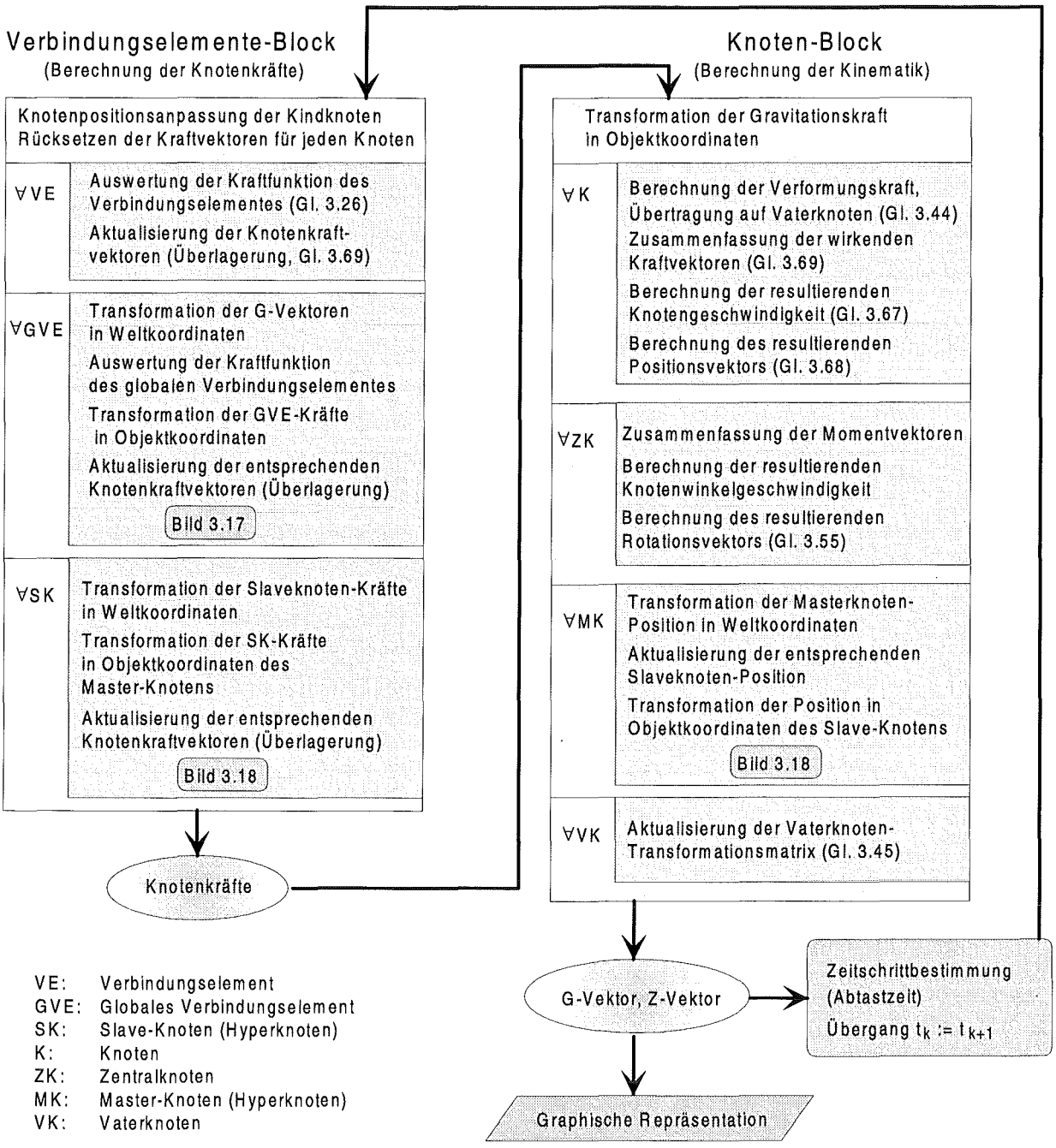


Bild 3.20: Ablaufschema des Berechnungsalgorithmus innerhalb MESD-Verfahrens

3.3.3 Stabilitätsbetrachtung

Ein Problem von numerischen Lösungsverfahren besteht generell darin, daß sich Genauigkeit und Geschwindigkeit der Berechnung umgekehrt proportional verhalten. Im allgemeinen ist mit der Genauigkeit auch die numerische Stabilität gekoppelt. Aufgrund der Echtzeitbedingung ist die Geschwindigkeit und Effizienz der Algorithmen in dieser Arbeit die wichtigere Anforderung. Trotzdem muß natürlich die Stabilität des Systems und der Verfahren gewährleistet bleiben.

Das MESD-Verfahren wurde so ausgelegt, daß es in sich ein sehr stabiles Verhalten zeigt, selbst bei größeren äußeren Anregungen. Neben der Modellbildungsstruktur und der Netz-Topologie ist auch das numerische Verfahren dieser Stabilitätsbedingung angepaßt. Notwendig sind jedoch sehr kleine Zeitschrittweiten im Bereich von Millisekunden. Unter diesen Voraussetzungen kann man sich auch eine Schrittweitenkontrolle ersparen, die die Effizienz der Algorithmen beeinträchtigen würde.

Bemerkung 3.4:

Für prinzipielle numerische Stabilität sollte die Abtastzeit und damit die Schrittweite der numerischen Integration h mindestens eine Größenordnung kleiner als die typischen Systemzeitkonstanten τ sein.

Zur Bestimmung der Systemzeitkonstanten werden die systemspezifischen Eigenfrequenzen herangezogen, die das dynamische Systemverhalten festgelegen. Hierzu muß die Systemmatrix A aus den Differentialgleichungen des deformierbaren Objektes abgeleitet werden, die dann zur Lösung der Eigenwertaufgabe verwendet wird [Foe90a]. Aufgrund der nichtlinearen und impliziten Struktur der Systemgleichungen (3.54) ist eine allgemeine Lösung nicht möglich. Um trotzdem Richtlinien zur Gewährleistung der Stabilität zu erhalten, wird an dieser Stelle ein sehr vereinfachtes Ersatzmodell eingeführt. Die Verhältnisse an einem Knoten werden hierbei in einer Momentaufnahme betrachtet, um den Ruhepunkt linearisiert und auf eindimensionale Kinematik übertragen. Damit kann eine Aussage über das grobe Systemverhalten in Abhängigkeit von den Systemparametern abgeleitet werden (Bild 3.21).

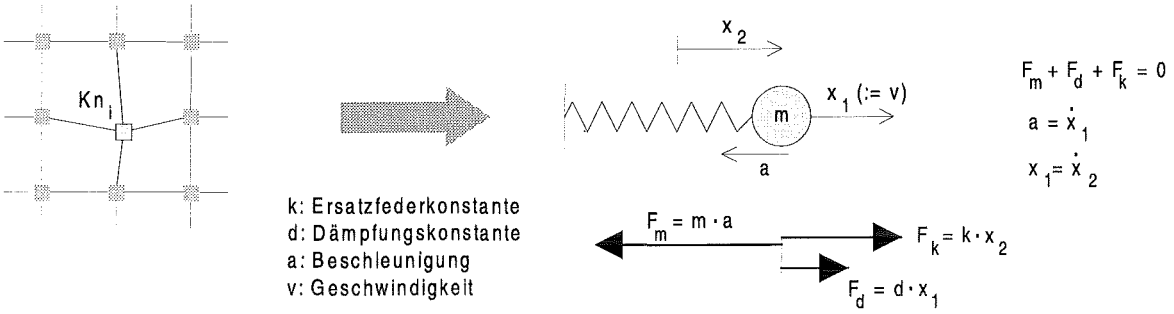


Bild 3.21: Systemvereinfachung zur Stabilitätsbetrachtung

Es wird vorausgesetzt, daß die Nachbarknoten im betrachteten Moment in Ruhe sind. Die VE-Ersatzkonstante k umfaßt die linearisierte Approximation des Knotenkraftvektors $\mathbf{f}$, also eine Kombination von mehreren VE-Parametern einschließlich der Verformungskraftkonstanten. Das entstehende System entspricht dem physikalischen Pendel, also einem gedämpften Masse-Feder-Schwinger. Die Systemmatrix lautet:

$$\mathbf{A} = \begin{bmatrix} -\frac{d}{m} & -\frac{k}{m} \\ 1 & 0 \end{bmatrix}, \qquad \dot{\mathbf{x}} = \mathbf{A} \cdot \mathbf{x} \tag{3.75}$$

Die Eigenwertbestimmung führt auf eine quadratische Gleichung:

$$det(\lambda \cdot \mathbf{I} - \mathbf{A}) = \lambda^2 + \frac{d}{m}\lambda + \frac{k}{m} = 0 \tag{3.76}$$

mit den Lösungen:

$$\lambda_{1/2} = -\frac{d}{2m} \pm \sqrt{\frac{d^2}{4m^2} - \frac{k}{m}} \tag{3.77}$$

Hierbei sind zwei Fälle[8] zu unterscheiden:

$$\frac{d^2}{4m^2} - \frac{k}{m} \begin{cases} > 0 \\ < 0 \end{cases} \qquad d > 2\sqrt{k \cdot m} \text{ (I)} \quad \vee \quad d < 2\sqrt{k \cdot m} \text{ (II)} \tag{3.78}$$

also einen Fall mit relativ großer Dämpfung (I) und einen Fall mit kleiner bis verschwindender Dämpfung (II). Wird $d, k > 0$ vorausgesetzt, liegt der Realteil der Eigenwerte stets links der imaginären Achse, das System ist somit stabil. Für die Systemzeitkonstanten $\tau = |1/\lambda|$ läßt sich festhalten:

1. Im Falle großer Dämpfung ($d >> 0$, Fall I) existieren zwei reelle Systemzeitkonstanten. Die geschlossene Lösung enthält somit keine harmonischen Anteile, das System reagiert auf eine Anregung mit einer aperiodischen Schwingung.

$$\tau_1 = \frac{2m}{d + \sqrt{d^2 - 4mk}}, \qquad \tau_2 = \frac{2m}{d - \sqrt{d^2 - 4mk}} \tag{3.79}$$

2. Im Falle kleiner Dämpfung ($d \to 0$, Fall II) existieren zwei konjugiert komplexe Eigenwerte und somit zwei getrennte Systemzeitkonstanten: die Dämpfungszeitkonstante τ_D (Realanteil) und die Schwingungszeitkonstante τ_S (Imaginäranteil). Das System reagiert auf eine Anregung mit einer gedämpften periodischen Schwingung.

$$\tau_D = \frac{2m}{d}, \qquad \tau_S = \frac{2m}{\sqrt{4mk - d^2}} \tag{3.80}$$

Aufgrund von Bemerkung 3.4 hat man mit Hilfe der Systemzeitkonstanten einen groben Anhaltspunkt, welchen Wert die Schrittweite h der numerischen Verfahren maximal erreichen darf. Für eine weitergehende Betrachtung muß die Abtasttheorie [Foe90b] hinzugezogen werden.

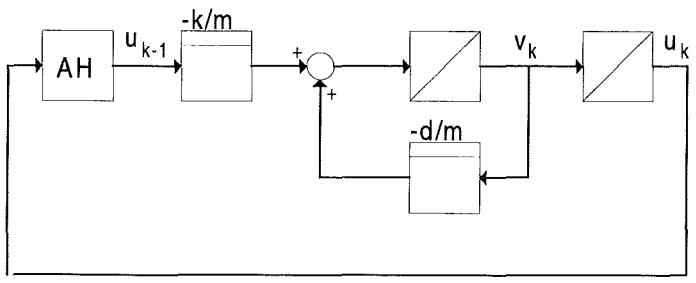


Bild 3.22: Blockstruktur des MESD-Lösungsalgorithmus

Wie oben gezeigt wurde, ist das System bei einer kontinuierlichen Betrachtung stets stabil. Mit der Simulation auf einem Digitalrechner geht jedoch eine zeitliche Diskretisierung einher, die neben den systemspezifischen Stabilitätsaspekten auch zusätzliche numerische Probleme verursacht. Deshalb soll das obige Einfach-System auch unter Verwendung der diskreten Lösungsalgorithmen des MESD-Verfahrens (Definition 3.24) betrachtet werden. Den zugrundeliegenden Regelkreis zeigt Bild 3.22, wobei zur Stabilitätsuntersuchung nur der innere Kreis ohne Führungsgröße berücksichtigt werden muß (Geschwindigkeit $v = x_1$, Auslenkung $u = x_2$).

[8] Der dritte Fall (Gleichheit, aperiodischer Grenzfall) wird hier nicht explizit betrachtet, er liefert keine zusätzlichen Ergebnisse bezüglich der Stabilität.

Für die resultierenden Integrationsgleichungen ergeben sich somit:

$$u_k = u_{k-1} + h \cdot v_k \,, \qquad v_k = v_{k-1} - \frac{d}{m} h \cdot v_k - \frac{k}{m} h \cdot u_{k-1} \qquad (3.81), (3.82)$$

Diese Differenzengleichungen lassen sich zum geschlossenen Kreis zusammenfassen:

$$u_k - u_{k-1}\left(1 - \frac{h^2 \frac{k}{m}}{1 + h\frac{d}{m}}\right) = \frac{h}{1 + h\frac{d}{m}} v_{k-1} \qquad (3.83)$$

An dieser Stelle erfolgt der Übergang in den Bildbereich, um die Stabilitätsgrenzen des Verfahrens zu bestimmen. Die Z-Transformation ergibt:

$$u(z)\left(1 - z^{-1}\left(1 - \frac{h^2 \frac{k}{m}}{1 + h\frac{d}{m}}\right)\right) = \frac{h}{1 + h\frac{d}{m}} z^{-1} \cdot v(z) \qquad (3.84)$$

Daraus läßt sich durch Umformung die Z-Übertragungsfunktion ableiten:

$$G(z) = \frac{v(z)}{u(z)} = \frac{\frac{h}{1 + h\frac{d}{m}}}{z - \left(1 - \frac{h^2 \frac{k}{m}}{1 + h\frac{d}{m}}\right)} \qquad (3.85)$$

Nach der Abtasttheorie [Foe90b] ist das diskretisierte dynamische System genau dann stabil, wenn alle Pole der Z-Übertragungsfunktion innerhalb des Einheitskreises der z-Ebene liegen. Es muß also gelten:

$$\left| 1 - \frac{h^2 \frac{k}{m}}{1 + h\frac{d}{m}} \right| < 1 \qquad (3.86)$$

An dieser Stelle ist eine Fallunterscheidung notwendig, ob der Betrag größer oder kleiner als 0 ist. Die Bedingung lautet:

$$1 - \frac{h^2 \frac{k}{m}}{1 + h\frac{d}{m}} \begin{cases} > 0 & \text{(I)} \\ < 0 & \text{(II)} \end{cases} \qquad (3.87)$$

Für die Dämpfung ergibt sich hierbei die Unterscheidung:

$$d > h \cdot k - \frac{m}{h} \quad \text{(I)} \qquad d < h \cdot k - \frac{m}{h} \quad \text{(II)} \qquad (3.88)$$

Im ersten Fall (große Dämpfung) ergibt sich die Bedingung:

$$-\frac{h^2 \frac{k}{m}}{1 + h\frac{d}{m}} < 0 \qquad (3.89)$$

Diese Ungleichung ist für alle d, $k > 0$ erfüllt. Der zweite Fall (kleine Dämpfung) liefert die Bedingung:

$$-(1-\frac{h^2\frac{k}{m}}{1+h\frac{d}{m}})<1 \tag{3.90}$$

Dies führt zu einer quadratischen Ungleichung

$$h^2-\frac{2d}{k}h-\frac{2m}{k}<0 \tag{3.91}$$

mit den Lösungen:

$$h_{1/2}=\frac{d}{k}\pm\sqrt{\frac{d^2}{k^2}+\frac{2m}{k}} \tag{3.92}$$

Da die Abtastzeit h stets positiv ist, folgt hieraus die Bedingung:

$$h<\frac{d}{k}+\sqrt{\frac{d^2}{k^2}+\frac{2m}{k}} \tag{3.93}$$

Aus Gleichung (3.88) ergibt sich für die Dämpfung weiterhin die Bedingung:

$$d>\frac{h\cdot k}{2}-\frac{m}{h} \tag{3.94}$$

Zusammengefaßt läßt sich für die Stabilität der MESD-Lösungsalgorithmen feststellen:

Bemerkung 3.5:

Der **Stabilitätsbereich** des MESD-Verfahrens läßt sich mit Hilfe eines einfachen Modells grob abschätzen. In einem lokalen Knotenbereich mit der Ersatzdämpfungskonstante d, der Ersatzfederkonstante k sowie der Masse m eines Knotens gilt:

Für eine Dämpfung $$d_S>\frac{h\cdot k}{2}-\frac{m}{h} \tag{3.95}$$

oder eine Zeitschrittweite $$h_S<\frac{d+\sqrt{d^2+2mk}}{k} \tag{3.96}$$

kann das nodale System eines deformierbaren Objektes numerisch stabil berechnet werden.

Diese Aussage kann durch Versuche bestätigt werden, wie im folgenden Unterkapitel gezeigt wird.

3.3.4 Vergleich numerischer Verfahren

In diesem Kapitel soll die Effizienz und Stabilität verschiedener numerischer Lösungsverfahren für Anfangswertprobleme 2. Ordnung untersucht werden. Hierzu werden drei typische deformierbare Objekte simuliert:

Modell 1: Geschlossener, kugelähnlicher Körper mit 130 G-Knoten und 1 Z-Knoten,
Parameter: Knoten m = 0,002 kg, m_Z = 0,22 kg, d = 0,2 Ns/m, s_V = 0,2 N/m,
lineare VE ohne Beschränkung, s_{VE} = 1,2 N/m

Modell 2: rohrähnlicher Körper mit 70 G-Knoten und 7 Z-Knoten,
Parameter: Knoten m = 0,002 kg, m_Z = 0,02 kg, d = 0,4 Ns/m, s_V = 0,5 N/m,
lineare VE ohne Beschränkung, s_{VE} = 0,2 N/m

Modell 3: Verknüpfung von Modell 1 & Modell 2 mit 10 Hyperknoten
(10 Masterknoten in Modell 1, 10 Slaveknoten in Modell 2)

Es werden sowohl einstufige als auch zweistufige Lösungsverfahren untersucht. Iterative und vielschrittige Methoden werden wegen der Echtzeitbedingung nicht betrachtet. Zur Herleitung der klassischen Verfahren wird auf [BS87] oder [Sch88] verwiesen.

1. *MESD-Verfahren* (einstufiges modifiziertes Euler-Cauchy-Verfahren) nach Definition 3.24

2. *Quasidynamisches Verfahren*

Diese Methode ist eine Abwandlung des MESD-Verfahrens für $m \to 0$. Hieraus resultiert eine vereinfachte Differentialgleichung 1. Ordnung (v ~ F). Bei großen Dämpfungen ist diese Vereinfachung zulässig, ohne die Charakteristik der Bewegung zu verfälschen.

$$\mathbf{v}_k = \frac{1}{d}(\mathbf{F}_{A,k} + f_k(\mathbf{x}_{k-1}, \dot{\mathbf{x}}_{k-1})) \tag{3.97}$$

$$\mathbf{u}_k = \mathbf{u}_{k-1} + \mathbf{B} \cdot h \cdot \mathbf{v}_k \tag{3.98}$$

3. *Modifiziertes Runge-Kutta-Verfahren* 2. Ordnung (mit impliziter Dämpfung)

Die Lösung wird hierbei in zwei Stufen berechnet, Grundlage für die endgültigen Lösungen sind die Zwischenlösungen v'_k, u'_k nach der Schrittweite *h*/2. Die Knotenkraftfunktionen müssen hierbei auch zweimal durchlaufen werden, es wird also im Vergleich zum MESD-Verfahren eine doppelte Rechenzeit benötigt.

$$\mathbf{v}'_k = \frac{1}{1 + \frac{d \cdot h}{2m}}(\mathbf{v}_{k-1} + \frac{h}{2m}(\mathbf{F}_{A,k} + f_k(\mathbf{x}_{k-1}, \dot{\mathbf{x}}_{k-1}))) \tag{3.99}$$

$$\mathbf{u}'_k = \mathbf{u}_{k-1} + \mathbf{B} \cdot \frac{h}{2} \cdot \mathbf{v}_k \tag{3.100}$$

$$\mathbf{v}_k = \frac{1}{1 + \frac{d \cdot h}{m}}(\mathbf{v}_{k-1} + \frac{h}{m}(\mathbf{F}_{A,k} + f_k(\mathbf{x}'_{k-1}, \dot{\mathbf{x}}'_{k-1}))) \tag{3.101}$$

$$\mathbf{u}_k = \mathbf{u}_{k-1} + \mathbf{B} \cdot h \cdot \mathbf{v}_k \tag{3.102}$$

4. *Modifiziertes Heun-Verfahren* (mit impliziter Dämpfung)

Das Heun-Verfahren ist ein Prädiktor-Korrektor-Verfahren, also auch zweistufig. Der vorläufigen Lösung v'_k, u'_k nach dem ersten Schritt folgt ein Korrekturschritt zur Berechnung der endgültigen Lösung. Die Knotenkraftfunktionen müssen ebenso zweimal durchlaufen werden. Die Berechnungsvorschriften für den ersten Schritt entsprechen den Gleichungen (3.67) und (3.68) des MESD-Verfahrens.

$$\mathbf{v}_k = \frac{1}{1 + \frac{d \cdot h}{2m}}(\frac{\mathbf{v}_{k-1} + \mathbf{v}'_k}{2} + \frac{h}{2m}(\mathbf{F}_{A,k} + f_k(\mathbf{x}'_{k-1}, \dot{\mathbf{x}}'_{k-1}))) \tag{3.103}$$

$$\mathbf{u}_k = \frac{\mathbf{u}_{k-1} + \mathbf{u}'_k}{2} + \mathbf{B} \cdot \frac{h}{2} \cdot \mathbf{v}_k \tag{3.104}$$

5. *Klassisches Euler-Cauchy-Verfahren* (ohne implizite Dämpfung)
   Die Lösung entspricht den Gleichungen (3.74) und (3.78).
6. *Klassisches Runge-Kutta-Verfahren* 2. Ordnung (ohne implizite Dämpfung)
   Die Berechnung von $\mathbf{u}_k$ erfolgt analog zu (3.100) und (3.102).

$$\mathbf{v}'_k = \mathbf{v}_{k-1}(1-\frac{d\cdot h}{2m})+\frac{h}{2m}(\mathbf{F}_{A,k}+f_k(\mathbf{x}_{k-1},\dot{\mathbf{x}}_{k-1})) \tag{3.105}$$

$$\mathbf{v}_k = \mathbf{v}_{k-1}(1-\frac{d\cdot h}{m})+\frac{h}{m}(\mathbf{F}_{A,k}+f_k(\mathbf{x}'_{k-1},\dot{\mathbf{x}}'_{k-1})) \tag{3.106}$$

7. *Klassisches Heun-Verfahren* (ohne implizite Dämpfung)

Der erste Schritt entspricht dem Euler-Verfahren, Gleichungen (3.74) und (3.68). Die Berechnung von $\mathbf{u}_k$ erfolgt analog zu (3.104).

$$\mathbf{v}_k = (\frac{\mathbf{v}_{k-1}}{2}+\frac{\mathbf{v}'_k}{2}(1-\frac{d\cdot h}{m})+\frac{h}{2m}(\mathbf{F}_{A,k}+f_k(\mathbf{x}'_{k-1},\dot{\mathbf{x}}'_{k-1}))) \tag{3.107}$$

Für die Bestimmung der Stabilitätsgrenze im Simulationsversuch wird der feste Zeitschritt h erhöht, bis im Modell Instabilitäten auftreten. Der eingestellte Wert wird als kritischer Zeitschritt h_{krit} bezeichnet. Die Anregung der nodalen Systeme erfolgt durch Aufbringen einer Kraft (beschränkte Rampenfunktion) an einen G-Knoten der Modelloberfläche. Um die verschiedenen Verfahren direkt vergleichen zu können, wird ein Stabilitätsfaktor σ_S eingeführt. Hierbei ist s_{NV} die Stufigkeit des jeweiligen Verfahrens, also ein Multiplikator für die Anzahl der Durchläufe und damit die benötigte Rechenzeit pro Gesamtberechnungsschritt. Hinsichtlich den Anforderungen dieser Arbeit kann σ_S als ein Gütemaß des numerischen Verfahrens angesehen werden (Effizienzkriterium).

$$\sigma_S = \frac{h_{krit}}{s_{NV}} \tag{3.108}$$

Tabelle 3.1: Stabilitätsfaktor σ_S [s] für verschiedene numerische Lösungsverfahren

	MESD (1)	QDYN (2)	M-RK2 (3)	M-HEUN (4)	EULER (5)	RK2 (6)	HEUN (7)
Stufigkeit	1	1	2	2	1	2	2
Modell 1	0,056	0,041	0,0285	0,028	0,0145	0,007	0,011
Modell 2	0,0595	0,032	0,0295	0,029	0,0095	0,0049	0,0072
Modell 3	0,051	0,0098	0,0285	0,028	0,0092	0,0048	0,0065

Die Auswertung zeigt, daß mehrstufige Verfahren keine Verbesserung des numerischen Stabilitätsverhaltens bieten. Die Methoden mit impliziter Dämpfung sind zur Echtzeitsimulation besser geeignet als klassische Verfahren, da sie einen wesentlich größeren Stabilitätsbereich erlauben. Das numerische Lösungsverfahren der MESD-Methode besitzt bei allen Modellen den größten Stabilitätsfaktor σ_S und ist deshalb zur Echtzeitsimulation nodaler Systeme sehr gut geeignet. Es soll jedoch noch einmal betont werden, daß diese Ergebnisse unter den spezifischen Anforderungen dieser Arbeit gelten. Insbesondere wurden Genauigkeitsaspekte zugunsten der Echtzeitfähigkeit vernachlässigt. Die aus dieser Arbeit hervorgegangenen numerischen Algorithmen (Definition 3.24) sind der Modellbildungsstruktur des MESD-Verfahrens angepaßt, andere Strukturen nodaler Netze können abweichende Ergebnisse hervorbringen [Deu96].

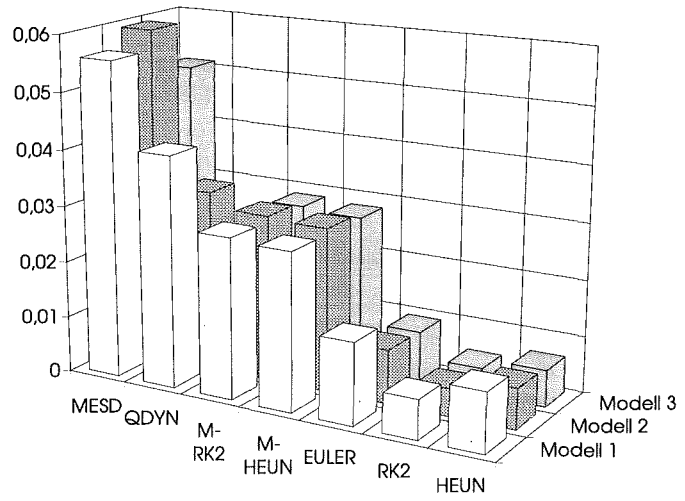


Bild 3.23: Graphische Darstellung des Stabilitätsfaktors σ_S für numerische Verfahren

3.4 Implementierung

In diesem Kapitel werden Aspekte der Software-Implementierung des MESD-Verfahrens erörtert. Dies umfaßt die Definition der deformierbaren Objekte innerhalb des Simulationssystems KISMET, die rechnerinterne Datenrepräsentation sowie Überlegungen zur Schrittweitenbestimmung und der Parallelisierung.

3.4.1 Objektprimitive

Die Definition der deformierbaren Körper erfolgt im KISMET-spezifischen Datenmodell, das Simulationsszenario ist hierbei durch eine hierarchische Dateiorganisation bestimmt [Kue91]. Die Dateien besitzen ein textuelles ASCII-Format und sind daher vom Benutzer einfach zu erzeugen und zu modifizieren. Im Rahmen der vorliegenden Arbeit wurde mit dem Progamm 'CREATER' ein Modellierwerkzeug zur benutzerfreundlichen Erstellung von MESD-Modellen geschaffen. Ein weiteres Programm 'TRIANG' dient zur Optimierung der Knotenanordnung und zur Erzeugung von Dreiecks-Maschennetzen, um die Geometriedaten für die Graphikhardware anzupassen und damit die Darstellungsgeschwindigkeit zu erhöhen.

Die Integration der Objekte in das Gesamtmodell erfolgt über eine spezifische Geometriedatei (Anhang B). Diese Definitionsdatei setzt sich aus dem graphisch-geometrischen Modell (Kapitel 5) und aus dem physikalischen Modell nach dem MESD-Verfahren zusammen. Der zweite Teil gliedert sich hierbei im wesentlichen in drei Komponenten:

- allgemeine Objektattribute: Standardwerte und Parameter zur physikalischen Modellbildung, Objekt- und Interaktionsverhalten
- Liste aller Masseknoten mit zugewiesenem Punkt des geometrisch-graphischen Modells sowie den Knotenattributen, Knotenbeziehungen
- Liste aller Verbindungselemente mit Knotenreferenzen, Charakteristiken und Parametern

Objektübergreifende Komponenten wie globale Verbindungselemente, Hyperknoten und globaldynamische Objekte werden in einer eigenen Datei des Gesamtmodells definiert ('ELADYN_HYPER', Anhang B.4).

Eine entsprechende Modellerstellung ist über die explizite Angabe und Spezifikation jedes Knotens und jedes Verbindungselementes möglich. Für den praktischen Einsatz können jedoch Grundstrukturen vorgegeben werden, die die Modellerstellung vereinfachen (Bilder 3.24-3.26). Diese Objektprimitive dienen sozusagen als Grundgerüst, das noch individuell erweitert und modifiziert werden kann. Die Spezifikation der Definitionsdateien ist im Anhang B ausführlich erläutert.

'ORGAN'

Geschlossenes Objekt (in beiden Oberflächenrichtungen)

Organisation: G-Knotenmatrix (s x t)
Alle G-Knoten besitzen gleiche Attribute

1 Z-Knoten: zentraler Vaterknoten für alle G-Knoten (globale Bewegung)

Verbindungselemente: Alle direkten G-Knoten-Nachbarn unmittelbar verknüpft (nach Topologie Bild 3.15C)

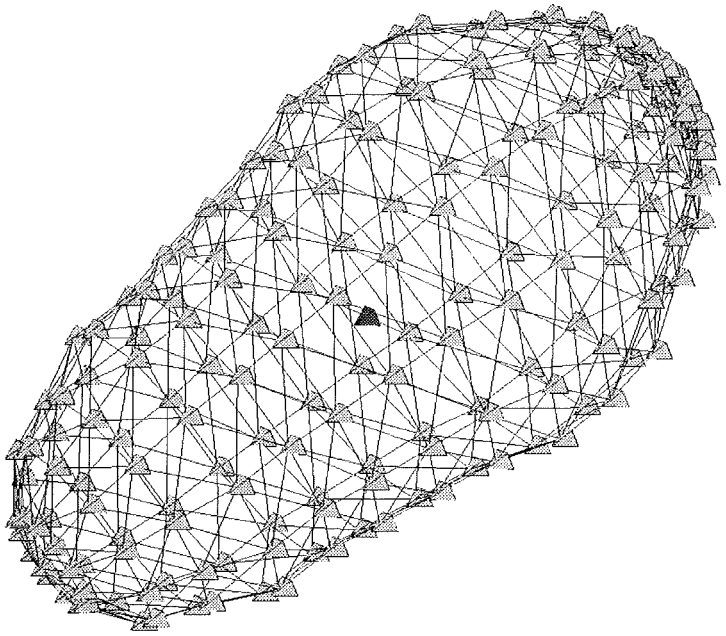


Bild 3.24: Objektprimitiv 'ORGAN'

'ROHR'

Teilgeschlossenes Objekt (in einer Oberflächenrichtung)

Organisation: G-Knotenmatrix (s x t)
Alle G-Knoten besitzen gleiche Attribute

(t) Z-Knoten: Gruppenweise Organisation von s G-Knoten mit einem Z-Knoten als Vaterknoten (partielle Verschiebung)

Verbindungselemente: Alle direkten G-Knoten-Nachbarn unmittelbar verknüpft (nach Topologie Bild 3.15C)
Erweiterte VE (Torsionsfeder-elemente und Biegefederelemente) für Z-Knoten

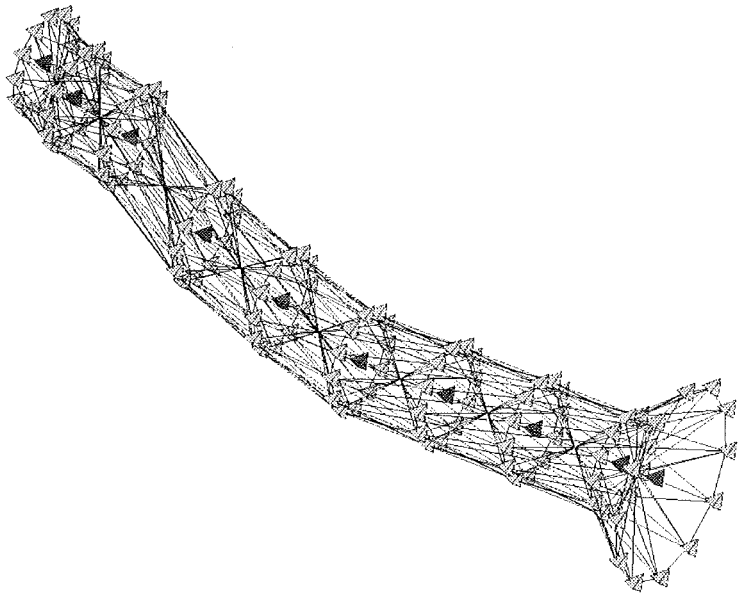


Bild 3.25: Objektprimitiv 'ROHR'

'GEWEBE'

Offenes Objekt

Organisation: G-Knotenmatrix (s x t)
Alle G-Knoten besitzen gleiche Attribute, keine Vaterknoten
Keine Z-Knoten

Verbindungselemente: Alle direkten G-Knoten-Nachbarn unmittelbar verknüpft (nach Topologie Bild 3.15C)

(dient zur Verknüpfung anderer Objekte, Bindegewebe)

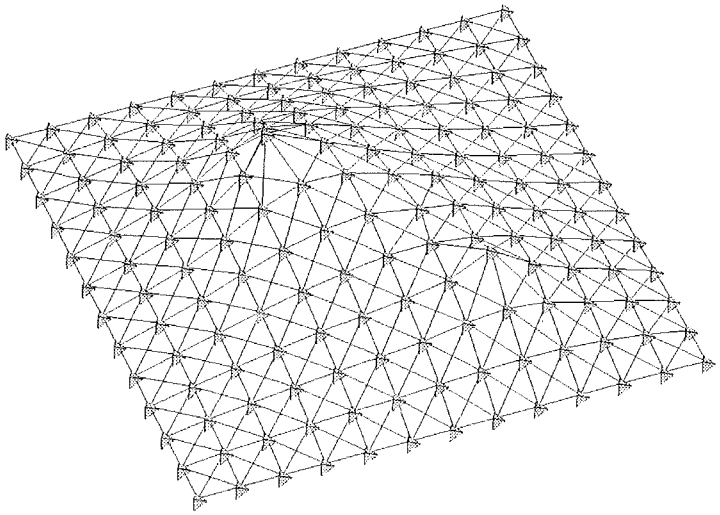


Bild 3.26: Objektprimitiv 'Gewebe'

3.4.2 Modulkonzept und Datenstruktur

Zentrales Ziel der Softwareimplementierung ist eine effiziente, echtzeitfähige Simulation. Hierzu gehört neben der entsprechenden Modellbildung und schnellen Algorithmen auch eine Minimierung des rechnerinternen Datenaustausches sowie eine effektive Datenhaltung. Hierbei sind auch stets die hardwarespezifischen Eigenschaften und Fähigkeiten zu berücksichtigen.

Analog zur Gesamtstruktur der Modellbildung (Bild 2.2, Bild 2.3) kann die Software in drei funktionell getrennte Module mit spezifizierten Schnittstellen unterteilt werden. Der Informationsfluß zwischen den Modulen soll hierbei möglichst klein gehalten werden (Bild 3.27). Das Dynamikmodul wirkt hierbei als Datenquelle ('Producer'), das Graphikmodul als Datensenke ('Consumer'). Die Datenstruktur soll auch hinsichtlich einer effizienten Modellmodifikation organisiert werden, da die Struktur des Modells bei manchen Manipulationen durch Hinzufügen und Entfernen von Komponenten verändert wird.

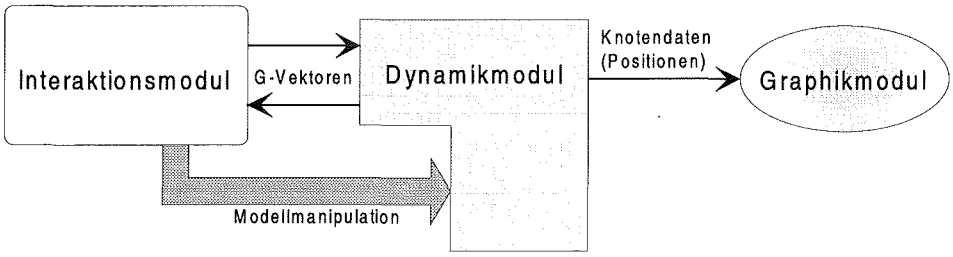


Bild 3.27: Modulkonzept der Softwareimplementierung

Ein entsprechender Ansatz ist die Auftrennung aller Knotendaten in drei funktionell getrennte Bereiche, die im Speicher jeweils zusammen allokiert werden und deshalb in sich einen homogenen, zusammenhängenden Speicherbereich bilden (Bild 3.28). Hiermit wird die Rechnerarchitektur unterstützt, da diese Bereiche geschlossen im schnellen Cachespeicher zur Verfügung gestellt werden können. Um auch bei größeren Modellmodifikationen noch

geschlossene Speicherbereiche zu gewährleisten, werden zunächst größere Speicherbereiche allokiert als für das Originalmodell benötigt werden.

Zentralbereich

Array von Knotenstrukturen mit allen Daten des physikalischen Modells, auf die nur vom Dynamikmodul zugegriffen wird. Die Liste ist zusätzlich mit Zeigern auf den Vorgänger- und Nachfolger-Knoten versehen (doppelt verkettete Liste), um das Einfügen und das Löschen von Knoten zu vereinfachen. Dies ist beispielsweise für die Implementierung eines Schneidevorganges notwendig (Kapitel 4). Weiterhin existieren Zeiger auf die entsprechenden Listeneinträge des Interaktionsbereichs und der Knotendomänen. Bild 3.28 zeigt die entsprechende strukturelle Anordnung für ein nodales Netz mit n Masseknoten (ohne Z-Knoten und F-Knoten).

Bereich der Knotendomänen

Hierin werden alle geometriespezifischen Daten gespeichert, die das Graphikmodul zur Darstellung des Objektes benötigt (aktuelle Knotenposition, Normalenvektor, Texturkoordinaten). Durch die kompakte Datenhaltung wird der Informationsaustausch zwischen Dynamik- und Graphikmodul beschleunigt.

Interaktions-Bereich

Hierin werden die Knotendaten verwaltet, die ausschließlich zur Behandlung von Interaktionen benötigt werden: Nachbarschaftsbeziehungen, Liste der Verbindungselemente und Facettenlisten.

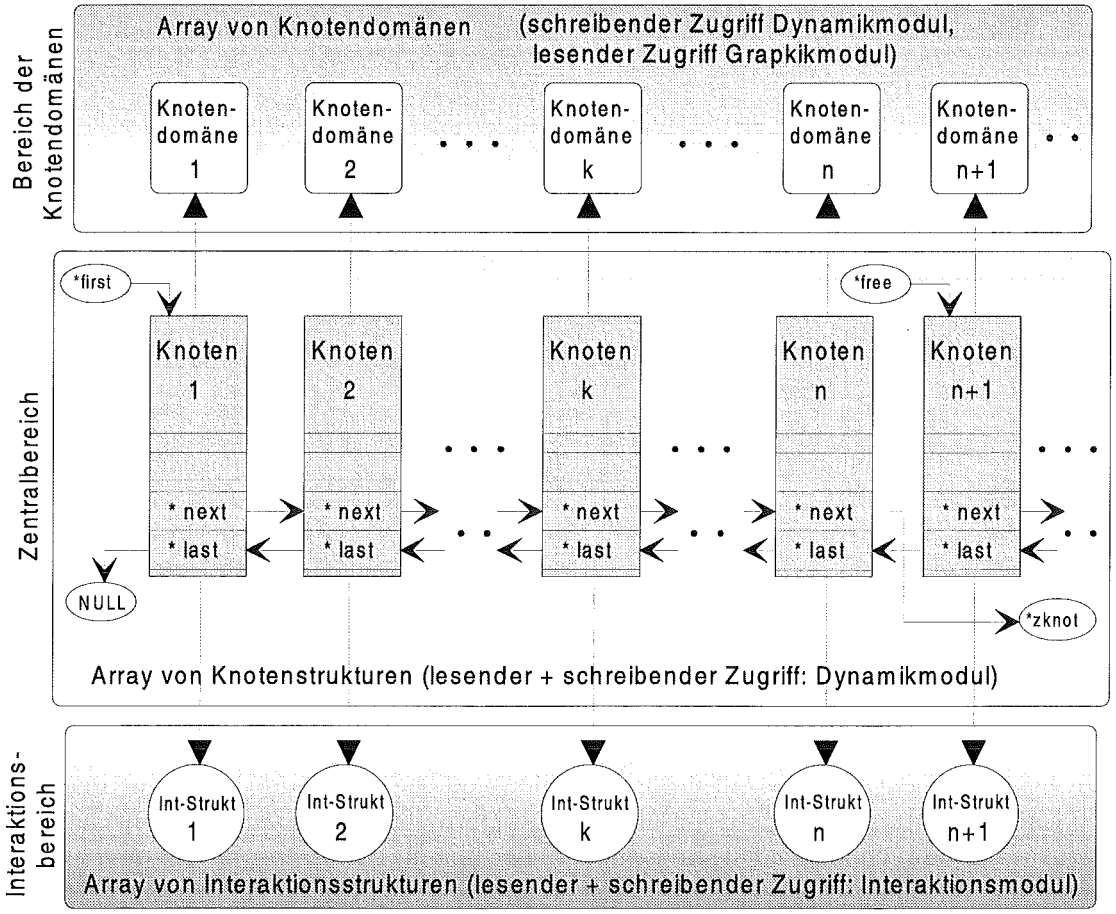


Bild 3.28: Schema der Datenstruktur des nodalen Modells (Knoten)

Die Datenlisten der Z-Knoten und F-Knoten sind ähnlich aufgebaut. Hier ist allerdings der Bereich der Knotendomänen überflüssig, da diese Knoten nicht graphisch ausgegeben werden. Die Liste der Verbindungselemente ist ebenfalls doppelt verkettet ausgeführt, um eine Neuorganisation während der Laufzeit zu erleichtern. Die Daten werden jedoch in einem geschlossenen Array verwaltet, da nur das Dynamikmodul zugreift.

3.4.3 Echtzeitkriterien und Parallelisierungsaspekte

Eine Simulation in Echtzeit impliziert, daß die gewählte Zeitschrittweite des numerischen Verfahrens mit der Durchlaufzeit für eine gesamte Simulationsschleife übereinstimmt. Ein Durchlauf beinhaltet neben den Modulen von Bild 3.27 (Dynamik, Interaktion, Graphik) auch weitere Simulationsaufgaben wie die Verwaltung der Benutzerschnittstellen, die gesamte Graphikaufbereitung sowie die Darstellung der starren Objekte des Simulationsszenario. Die Intervallzeitschritte werden mit Hilfe einer Echtzeituhr vor jedem Berechnungszyklus neu bestimmt ($h = T_{i+1} - T_i$), das dynamische Verhalten ist somit unabhängig von Nebenbedingungen wie Rechnerleistung und Modellgröße. Die Zeitschrittweite h darf jedoch nicht über eine kritische Schrittweite h_{krit} steigen, um die numerische Stabilität nicht zu gefährden (Kapitel 3.3). Neben der Verringerung des Modellumfangs trägt hierzu eine geeignete Aufteilung der zur Verfügung stehenden Prozessorrechenzeit bei.

Das Dynamikmodul kann für jedes deformierbare Objekt n-fach durchlaufen werden, bevor die graphische Verarbeitung und sonstige Aufgaben durchgeführt werden. Innerhalb dieser inneren Dynamik-Schleife verringert sich die Zeitschrittweite auf $h=h'/n$. Die Bildwiederholrate sinkt entsprechend, bei 10-15 Bildern/s ist die interaktive Grenze erreicht.

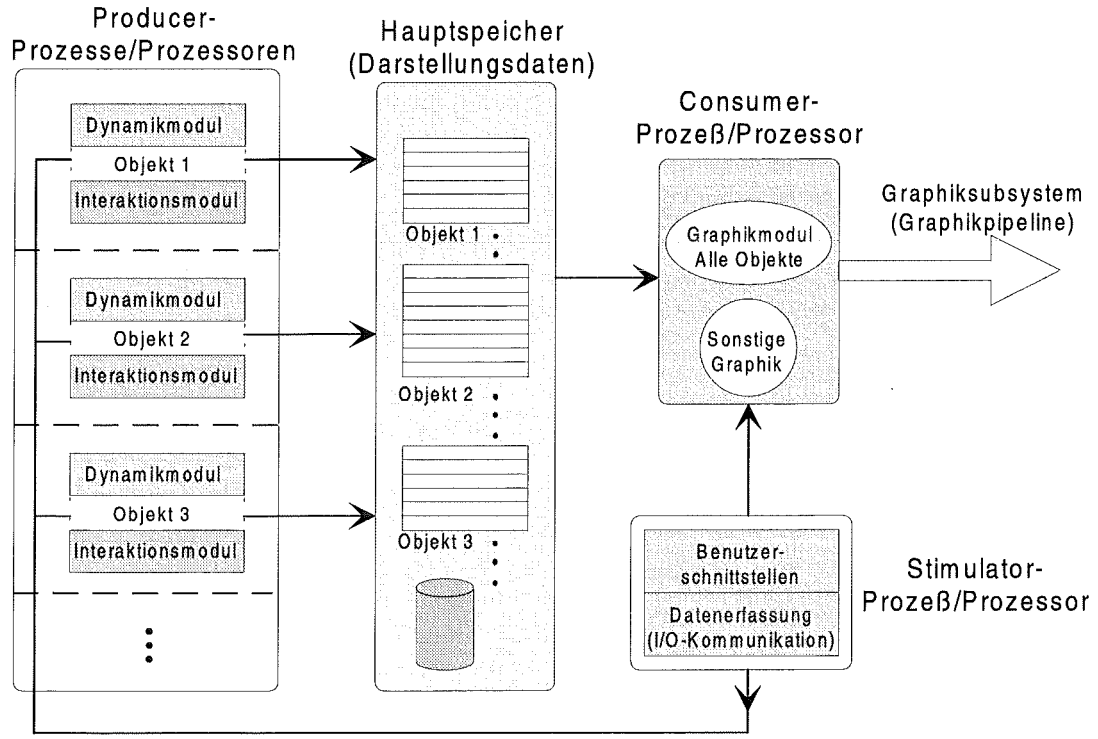


Bild 3.29: Prozeßverteilung bei Parallelisierung

Eine weitere, sehr effektive Möglichkeit bietet die Nutzung von mehreren Prozessoren auf einem entsprechenden Rechnersystem. Die in dieser Arbeit zur Verfügung stehende

Hardwareplattform (Silicon Graphics *Onyx*) besitzt zwei Hauptprozessoren, kann aber optional auf bis zu 16 Prozessoren aufgerüstet werden.

Die für die Dynamik und die Interaktion zuständigen Module werden auf einen zweiten Prozessor ausgelagert. Beide Prozesse sind bezüglich der Programmabläufe entkoppelt und nicht synchronisiert. Dies gewährleistet neben einer höchstmöglichen Bildwiederholrate auch eine Stabilitätssicherung unabhängig vom Aufwand der graphischen Ausgabe. Der eine Prozeß führt die dynamischen Berechnungen auf Basis der Objektparameter durch und manipuliert hierbei die Geometriedaten, er produziert sozusagen die Darstellungsdaten (Datenquelle, 'Producer'). Der andere Prozeß holt diese Geometriedaten ab, verarbeitet diese nach bestimmten Randbedingungen (Ansicht, Darstellung) und gibt sie in graphischer Form auf dem Bildschirm aus; er konsumiert die Daten (Datensenke, 'Consumer'). Bei bestimmten Benutzereingaben und Modelländerungen muß der Producer zur Sicherung der Datenkonsistenz gestoppt werden. Der Hauptspeicher dient als Schnittstelle zwischen den einzelnen Prozessen, über ihn werden die Darstellungsdaten ausgetauscht[9].

Stehen mehrere Prozessoren zur Verfügung, kann eine weitere objektbezogene Prozeßverteilung durchgeführt werden. Die Berechnung der einzelnen deformierbaren Körper kann unabhängig voneinander erfolgen, eine Synchronisation ist bei genügend kleinen Schrittweiten nicht notwendig. Das Graphikmodul sollte auf einem Prozessor vereint bleiben, da die Prozessor- und Graphikleistung aufeinander abgestimmt ist. Außerdem wäre ansonsten der Synchronisationsaufwand sehr groß. Der Prozeß der Datenerfassung (Simulationsanregung, 'Stimulator'), der die Verwaltung der Benutzerschnittstellen und Bediensysteme beinhaltet, kann ebenfalls auf einen Prozessor ausgelagert werden.

3.5 Aspekte zur Objektapproximation, Validierung

Zum Ende dieses Kapitels sollen noch einige Überlegungen durchgeführt werden, wie die Eigenschaften eines gegebenen Objektes vom MESD-Modell optimal approximiert werden können. Die Entwicklung von Verfahren ist jedoch nicht Ziel dieser Arbeit, es sollen jedoch Lösungsvorschläge aufgezeigt werden.

Innerhalb des MESD-Verfahrens lassen sich im Rahmen der Modellbildung folgende Systemkomponenten variieren, um eine gute Approximation des zu repräsentierenden Körpers zu erzielen:

- Verteilung der Knoten, Punktpositionierung
- Massenverteilung, Zuweisung der Knotenmassen und Dämpfungen
- Verknüpfung der Knoten, Netztopologie, Knotenbeziehungen (Vater-Kind-Knoten)
- Charakteristik und Parameter der Verbindungselemente

Zu einem Teil dieser Punkte gibt es schon verschiedene interessante Lösungsvorschläge. Deussen [Deu96] beschäftigt sich in seiner Arbeit mit der Fragestellung nach einer möglichst optimalen Punktpositionierung nodaler Netzmodelle. Er verwendet hierzu ein Voronoi-Iterationsverfahren, das auf Basis stochastisch verteilter Punktmengen operiert. In der gleichen Arbeit behandelt Deussen auch die Ermittlung einer Massenverteilung deformierbarer Körper aus einem Vergleich der resultierenden Massenmomente.

In der Finite-Elemente-Methode existieren schon Optimierungsverfahren zur Bestimmung und Integration von Objektkomponenten und -parametern [KW91]. Ausgehend von Referenz-

[9] Interprozeß-Kommunikation über einen gemeinsamen Speicherbereich ('Shared Memory').

verformungen, die über die gegebenen elastodynamischen Eigenschaften berechnet werden, sind über probabilistische Optimierungsverfahren wie 'Simulated Annealing' gute Approximationen erzielbar [DKT95]. In einer Folge von Versuchen werden Systemparameter verändert und das Gesamtverhalten des Systems beobachtet. In Abhängigkeit des Optimierungsalgorithmus werden die Veränderungen übernommen und neue Modifikationen durchgeführt.

Die Validierung stellt gewissermaßen die Rückkopplung über alle Schritte der Modellbildung und Simulation dar. Sie beantwortet die Frage nach der Gültigkeit des Simulationsmodells, das heißt, inwieweit es das Verhalten des Originalsystems im Hinblick auf die Aufgabenstellung richtig wiedergibt. Neben den qualitativen Aspekten kann auch die quantitative Güte berücksichtigt werden. Die Simulationsergebnisse werden hierbei mit Daten aus Versuchen am echten System verglichen.

Im Hinblick auf das Anwendungsfeld dieser Arbeit (Kapitel 6) wurde eine empirische Abschätzung der Modellparameter und Validierung durchgeführt. In einem iterativen Vergleichsprozeß erfolgte die Anpassung des Modells an das subjektive Verhalten von realem Gewebe. Hierbei sind allerdings die spezifischen Anforderungen zu berücksichtigen: für eine Trainingsumgebung in der Medizin sind keine exakten Gütekriterien einzuhalten, zumal solche auch nur unzureichend beschrieben werden können. In Kapitel 6.2.3 wird ausführlicher darauf eingegangen.

Zu unterscheiden ist die Validierung vom Prozeß der Verifikation, der bei der Modellbildung stets durchgeführt werden muß. Hierbei wird die Frage nach der systematischen Korrektheit eines Schrittes in Bezug auf die Vorgaben beantwortet. Im Mittelpunkt steht die Plausibilität und das qualitative Systemverhalten. Diese Kontrolle des Modells erfolgt nach jedem einzelnen Modellbildungsschritt:

- Gültigkeit der Verwendung physikalischer Gesetzmäßigkeiten zum Erstellen des physikalischen Modells
- Umsetzung in das mathematische Modell, Korrektheit und Anwendbarkeit der numerischen Methoden
- Implementierung zum Rechnermodell, beinhaltet Softwareentwurf und Programmtest

3.6 Zusammenfassung des Kapitels

Dieses Kapitel beschäftigte sich mit der physikalischen Repräsentation deformierbarer Objekte. Ausgehend von einer allgemeineren Beschreibung deformierbarer Körper wurde die Problematik und Anforderungsspezifikation aufgezeigt. Die Methode der nodalen Systeme wurde als Vereinfachung der allgemeinen Finite-Elemente-Methodik vorgestellt. Auf dieser Basis wurde das MESD-Verfahren eingeführt und ausführlich erläutert. Neben den Modellbildungskonzepten wurden die Aspekte der numerischen Simulation ausführlich betrachtet.

Mit dem vorgestellten universellen Ansatz ist es erstmals möglich, die mechanischen Eigenschaften deformierbarer Körper in Echtzeit zu simulieren. Die Modellbildung kann beliebige nichtlineare Charakteristiken berücksichtigen, es bestehen keine prinzipiellen Einschränkungen bezüglich der Art der Verbindungselemente. Das hierarchische Konzept deckt verschiedene generelle Bewegungstypen ab. Die Verknüpfung verschiedener Objekte ist möglich. Die Struktur und Eigenschaften der Modelle sind zur Simulationslaufzeit modifizierbar. In Kombination mit den entsprechenden numerischen Lösungsalgorithmen ermöglicht das MESD-Verfahren eine sehr schnelle und stabile Bewegungssimulation.

4 Interaktion und Manipulation deformierbarer Objekte

Eine wichtige Eigenschaft einer 'Virtual Reality'-Umgebung ist die Möglichkeit des interaktiven Eingreifens des Menschen in das synthetisch generierte Szenario. In diesem Kapitel wird die direkte Beeinflussung von deformierbaren Objekten auf Basis des MESD-Verfahrens betrachtet. Die Interaktion bewirkt hier eine Stimulation des physikalischen Modells und somit die Anregung der Simulation. Die Methoden, die in diesem Kapitel vorgestellt werden, sind für den MIC-Trainer optimiert, jedoch auch für andere Einsatzgebiete adaptierbar.

4.1 Anforderung und Konzeption

Die Interaktion in einer dreidimensionalen Simulationsumgebung mit Berücksichtigung der physikalischen Eigenschaften deformierbarer Objekte umfaßt folgende Teilaufgaben:

- *Kollisionserkennung*
  Diese beinhaltet die korrekte Erfassung des Zusammentreffens zweier Objekte, also eine geometrische Überprüfung auf einen gemeinsam belegten Teilraum im dreidimensionalen Szenario. Neben der Detektion soll auch eine Bestimmung der Kollisionsorte an den Objekten erfolgen.
- *Interaktionsauswertung*
  Die Interaktionsauswertung definiert die Reaktion des interagierenden Objekts und die Auswirkung auf das Modell. Diese Modellmanipulation betrifft in erster Linie die morphologische Änderung der Geometrie (Deformation). Je nach Art der Kollision, Typ der kollidierenden Teile und Eigenschaften des deformierbaren Objektes muß die Wechselwirkung unterschiedlich erfolgen.
- *Modellmodifikation*
  Eine weitergehende Manipulation erfordert neben der Verformung auch eine unterlagerte Modifikation des deformierbaren Objektes. Dies umfaßt die strukturelle Änderung des MESD-Modells einschließlich der Neuorganisation der Topologie und der Redefinition der Parameter und Attribute. Mit Hilfe von Modifikationsregeln wird hierbei ein spezifisches Verhaltensmodell generiert.

Angestrebt wird ein möglichst realistisches Interaktionsverhalten unter der strengen Anforderung der Echtzeitfähigkeit. Da nicht nur eine reine Kollisionserkennung, sondern auch eine Reaktion des deformierbaren Objektes auf die Manipulation erfolgt, ist die Interaktionsbehandlung relativ aufwendig. Es wird daher ein Konzept verfolgt, das auf hierarchisch gegliederten und effizienten Teilmodulen basiert. Prinzipiell wird die Nutzung möglichst effektiver Vereinfachungen und Abstraktionen angestrebt, ohne jedoch den angestrebten Realismus im Sinne der Aufgabenstellung in Frage zu stellen. Wichtig ist eine entsprechende Datenstruktur, deren Organisation die Modellmodifikationen in Echtzeit erlaubt (siehe auch Kapitel 3.4.2 sowie Bild 3.27). Da nicht alle Manipulationseigenschaften direkt mit physikalischen Gesetzmäßigkeiten beschreibbar sind, muß das MESD-Modell durch ein Verhaltensmodell erweitert werden, das aufgrund von Regelsätzen das Objektverhalten spezifiziert[1]. Die einzelnen Interaktionsmodule werden in den folgenden Teilkapiteln erläutert.

4.2 Effiziente Kollisionserkennung

Die Echtzeitanforderung stellt wiederum die wichtigste Nebenbedingung dar. Ein gängiges Verfahren in der Computergraphik besteht darin, zur Detektion von gegenseitigem Eindringen bei B-REP-Modellen alle Geometriefacetten des einen Körpers gegenüber allen Facetten des zweiten Körpers einzeln auf räumliche Überschneidung zu testen. Diese Methodik kann bei der vorliegenden Aufgabenstellung aufgrund der hohen Rechenzeit nicht übernommen werden. Ein weiteres Problem bei der Kollisionsdetektion besteht in der mehrfachen Diskretisierung der deformierbaren Objekte und der Simulation. Dies erschwert die korrekte Erfassung von gegenseitigem Eindringen der nodalen Modelle.

Räumliche Diskretisierung: Die Geometrie des deformierbaren Objektes wird aufgrund der physikalischen Modellbildung durch eine endliche Anzahl von diskreten Punkten repräsentiert (G-Knoten, nodales Modell). Das ursprüngliche Volumenobjekt wird also durch ein dreidimensionales Netz von eindimensionalen Punkten abstrahiert, das die Oberfläche des Körpers nachbildet. Die geometrische Form der Oberfläche zwischen den G-Knoten ist innerhalb des MESD-Modells nicht bestimmt.

Zeitliche Diskretisierung: Die die Dynamik bestimmenden Differentialgleichungen werden zur Simulation in Differenzengleichungen transformiert. Die Positionen der G-Knoten werden von numerischen Lösungsverfahren zu diskreten Zeiten ausgewertet. Für die Bewegung zwischen den festgelegten Simulationszeitpunkten kann keine konkrete Aussage getroffen werden.

Die Kollisionserkennung muß trotz dieser Problematik eine sichere Erfassung der Interaktionen ermöglichen. Hierzu werden abstrahierende Modellvereinfachungen durchgeführt. Dies bedingt eine prinzipielle Trennung von graphischer Repräsentation und Interaktionsbehandlung - es werden jeweils verschiedene, den Anforderungen angepaßte Modelle herangezogen.

Bemerkung 4.1:

Kollisionsprüfungen werden wesentlich vereinfacht und beschleunigt, wenn die zu untersuchenden Modelle aus einfachen geometrischen Grundformen aufgebaut sind (Quader, Zylinder, Kugel). Es ist deshalb vorteilhaft, mindestens einen der an der Interaktion beteiligten Körper mit diesen einfachen Geometrieprimitiven zu beschreiben.

[1] Man kann hier auch von einer 'Wissensbasis' sprechen. Im Gegensatz zu Expertensystemen sind jedoch in der vorliegenden Arbeit die Wissensbasis und die Inferenz-Komponente nicht getrennt implementiert.

4.2.1 Vereinfachung der Instrumentengeometrie

Betrachtet man die vorliegende Problemstellung, so erfolgt eine Manipulation deformierbarer Objekte meist über simulierte technische Systeme, die vom Menschen direkt gesteuert werden[2]. Diese Systeme sind als starre Modelle im Simulationsszenario eingebettet - im Falle der MIC-Anwendungen repräsentieren sie die chirurgischen Instrumente. Bei den nachfolgenden Ausführungen soll eine Kollision eines Instrumentes mit einem deformierbaren Objekt vorausgesetzt werden.

Bild 4.1 zeigt ein typisches Instrument für die Minimal-Invasive Chirurgie. Die Geometrie eines solchen Systems ist beliebig komplex, die Funktionalität jedoch relativ einfach. Bei näherer Betrachtung ergibt sich eine Möglichkeit zur Vereinfachung der Kollisionsprüfung durch Abstraktion der geometrischen Form. Das charakteristische MIC-Instrument besteht neben dem Griffstück, das für die Interaktion irrelevant ist, aus einem langen, geraden Schaft sowie dem Effektor. Der Effektor besitzt je nach Instrumententyp verschiedene Ausprägungen, besteht im allgemeinen jedoch aus ein oder zwei beweglichen Maulteilen. Somit kann ein Instrument durch 4 repräsentative Punkte symbolisiert werden: den Schaftursprung, den Effektoransatz sowie die beiden Effektorspitzen.

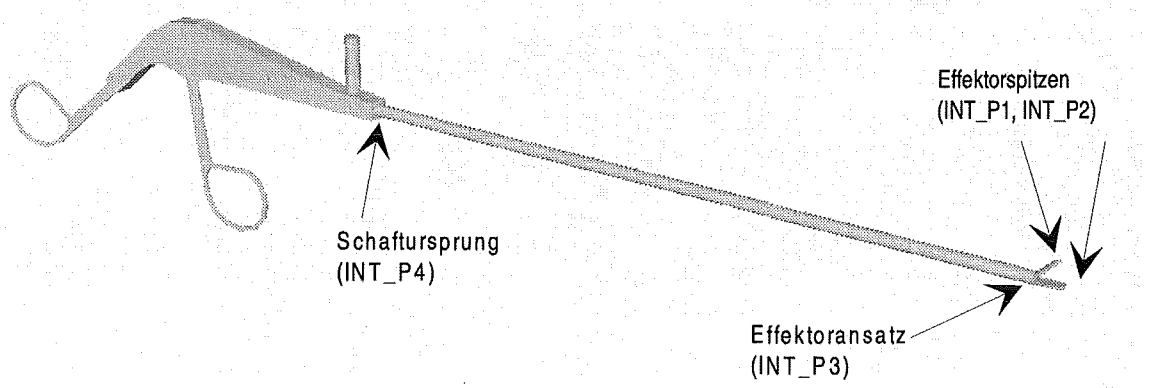


Bild 4.1: Modell eines typischen MIC-Instruments (Greifer)

Hieraus lassen sich drei charakteristische Strecken ableiten: die Schaftbasis (Schaftachse) sowie die beiden Effektorbasen. Um diese Strecken sind jeweils zylindrische Hüllbereiche definierbar. Das so entstehende Ersatz-Instrument basiert auf einer geometrischen Modellvereinfachung hinsichtlich der Kollisionsdetektion, während die funktionellen Eigenschaften des Instrumentes erhalten bleiben. Verallgemeinert läßt sich somit festhalten:

Bemerkung 4.2:

Eine effiziente Kollisionserkennung ist durch **abstrahierende Vereinfachung** des manipulierenden Systemmodells möglich. Das System wird hierzu in einen Satz von charakteristischen Strecken aufgeteilt, den sogenannten **Interaktionsstrecken**. Die Gesamtheit der Interaktionsstrecken mit jeweiligen zylindrischen Hüllvolumen bildet das Ersatz-System, das zur Kollisionsdetektion herangezogen wird.

[2] In KISMET werden diese Systeme allgemein mit 'Roboter' bezeichnet, da das Simulationssystem aus der Handhabungstechnik entstammt. Auch die MIC-Instrumente werden innerhalb KISMET als Roboter definiert.

Für eine grobe Kollisionsaussage werden die Interaktionshüllzylinder zu einem zylindrischen Volumen zusammengefaßt, der die möglichen interagierenden Teile des Instrumentes insgesamt umhüllt (Instrumentenhüllzylinder). Die Bestimmung aller Hüllvolumen wird bei der Initialisierung der Simulation durchgeführt und als zusätzliche Instrumentenattribute abgelegt.

Im allgemeinen sind den beiden kollidierenden Körpern unterschiedliche Objektkoordinatensysteme zugrundegelegt. Da die Instrumente kinematisch definiert sind und die Auslenkung ihrer Freiheitsgrade vom Bediener direkt vorgegeben werden, muß vor jeder Kollisionsprüfung eine Koordinatentransformation durchgeführt werden. Hierzu reicht es aus, die charakteristischen Instrumentenpunkte in das Koordinatensystem des zu überprüfenden deformierbaren Objekts zu transformieren[3].

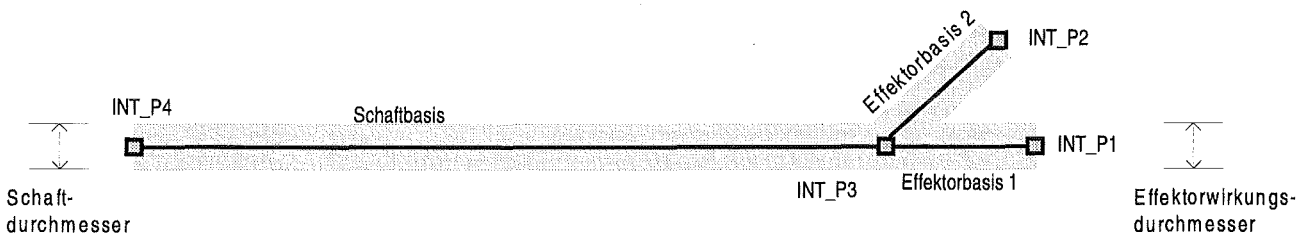
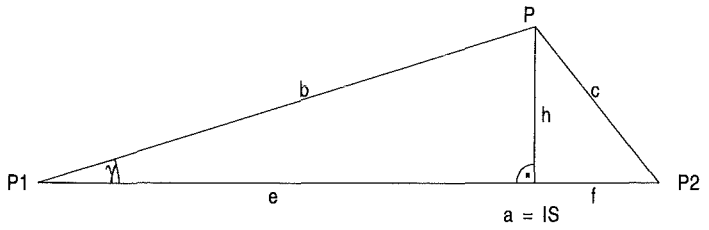


Bild 4.2: Ersatz-Instrument für Interaktionsdetektion

4.2.2 Geometrische Kollisionsprüfungen

Die in diesem Unterkapitel angewendeten geometrischen Prüfungen lassen sich auf einen einfachen und schnellen Triangulationstest zurückführen. Hierbei wird die Position eines Punktes P im dreidimensionalen Raum relativ zu der Interaktionsstrecke IS bestimmt, wobei IS durch die beiden Raumpunkte P1 und P2 definiert ist (Bild 4.3).



P	Position des Testpunktes im Raum
P1, P2	Charakteristische Streckenpunkte
IS (= a)	Interaktionsstrecke (Basis)
h	Abstand des Punktes von der Basis
e	Projezierter Basisabschnitt

Bild 4.3: Geometrischer Triangulationstest

Aus der Trigonometrie ergeben sich in einem Dreieck die Beziehungen:

$$\cos\gamma = \frac{e}{b}, \qquad c^2 = a^2 + b^2 - 2ab \cdot \cos\gamma \qquad (4.1), (4.2)$$

Eingesetzt und entsprechend aufgelöst ergibt sich der projezierte Basisabschnitt e sowie der Punktabstand h von der Interaktionsstrecke IS:

$$e = \frac{a^2 + b^2 - c^2}{2a}, \qquad h = \sqrt{b^2 - e^2} \qquad (4.3), (4.4)$$

[3] Innerhalb des KISMET-Geometriemodells sind diese Instrumentenpunkte als 'Workframes' definiert

Zur Bestimmung des Einflußbereichs erfolgt meist ein Vergleich des Abstandes h mit einer definierbaren Schranke. Diese Interaktionsschranke s_I leitet sich je nach Überprüfung aus einem Hüllzylinderradius, einem Hüllkugelradius oder aus einer entsprechenden Summe ab.

$$h < s_I \quad \text{oder} \quad h^2 = b^2 - e^2 < s_I^2 \qquad (4.5), (4.6)$$

Für eine effiziente Auswertung ist Gleichung (4.6) vorzuziehen, da hierdurch das aufwendige Berechnen der Wurzelfunktion für jeden getesteten Punkt entfällt. Die Quadratur des Schrankenwertes ist dagegen nur einmal notwendig.

Das implementierte Konzept beruht auf einem hierarchischen Prinzip, um die Rechenzeit der Überprüfungen zu minimieren. Jeder Schritt wird nur durchgeführt, wenn der vorangegangene erfolgreich war.

1. Hüllvolumentest des gesamten Instruments

Zuerst wird geprüft, welche der vorhandenen deformierbaren Objekte für eine eventuelle Interaktion mit dem Instrument in Frage kommen. Für ein Objekt wird bei der Initialisierung eine Hüllkugel um jeden Z-Knoten definiert[4], die sogenannte Z-Hüllkugel (Bild 4.4). Der Z-Knoten dient als Hüllkugelzentrum, der Abstand zum entferntesten zugehörigen Kindknoten bestimmt den Hüllkugelradius. Der Test ist positiv, wenn eine der Hüllkugeln einen minimalen Abstand zum Hüllzylinder des Instruments unterschreitet.

2. Hüllvolumentest der einzelnen Interaktionsstrecken

Die Überschneidung der einzelnen Streckenhüllzylinder des Instruments mit den Z-Hüllkugeln wird geprüft. Der Test ist erfolgreich, falls ein Volumenpaar einen definierbaren minimalen Abstand unterschreitet.

3. Bereichstest der zugehörigen G-Knoten

Für die G-Knoten, die den positiv getesteten Z-Hüllkugeln zuzuordnen sind, erfolgt nunmehr eine Überprüfung, ob diese im Einflußbereich der charakteristischen Interaktionsstrecken liegen. Das Unterschreiten eines einzustellenden Schrankenwertes hat einen positiven Testausgang zur Folge.

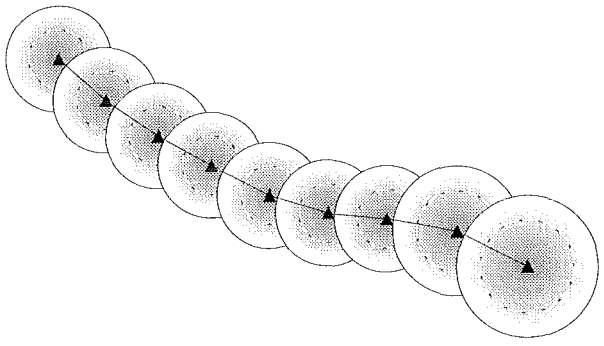


Bild 4.4: Z-Hüllkugeln bei Objektprimitiv 'Rohr'

Diese drei Schritte der geometrischen Kollisionsprüfung sind Näherungstests. Sie liefern als Ergebnis diejenigen G-Knoten, die im Wirkungsbereich einer Interaktionsstrecke sind. Die Einstellung der Schrankenwerte (s_I) des Triangulationstests muß in Abhängigkeit der Güte (und damit Größe) des MESD-Modells erfolgen: je besser die Modellgüte, desto kleiner der Abstand zwischen den einzelnen G-Knoten und desto kleiner können die Schranken gewählt werden. Zu große Schranken verursachen unnötige Prüfungen und verringern dadurch die Effizienz.

[4] Existiert zu einem Objekt kein Z-Knoten (wie bei dem Objektprimitiv 'Gewebe'), so wird ein anderer, zentral angeordneter Knoten als Hüllkugelzentrum definiert

4.2.3 Plausibilitätsprüfung

Die positiv getesten G-Knoten müssen einer weiteren Prüfung unterzogen werden, die insbesondere aufgrund der morphologischen Variabilität der deformierbaren Objekte notwendig ist. Die bisher durchgeführten geometrischen Prüfungen erlauben keine exakte Aussage über die Auswirkung der Kollision, da nur der räumliche Einfluß von Interaktionshüllvolumen auf diskrete Punkte der Geometrie (G-Knoten) untersucht wurde. Eine direkte Wechselwirkung tritt jedoch erst dann ein, wenn ein 'sinnvoller' Kontakt erfaßt wird, wenn also der G-Knoten von der Interaktionsstrecke unmittelbar beeinflußt wird. Aufgrund der oben angesprochenen Diskretisierungsproblematik können ansonsten unplausible Effekte wie das 'Ankleben' und 'Springen' von G-Knoten an das Instrument auftreten. Die Abbildungen 4.5 und 4.6 verdeutlichen dies, wobei die Interaktionsstrecke IS jeweils senkrecht zur Bildebene angeordnet ist. Links ist das Problem der zeitlichen Diskretisierung aufgeführt, zwischen zwei aufeinanderfolgenden Simulationszeitpunkten ist die Interaktionsstrecke durch einen G-Knoten durchgedrungen. Das Problem der räumlichen Diskretisierung zeigt das rechte Bild, die Interaktionsstrecke dringt zwischen zwei G-Knoten in das Objekt ein. Die Plausibilitätsprüfung muß diese Fälle erkennen und die entsprechenden Kollisionen detektieren, um die korrekten Modellmanipulationen einleiten zu können.

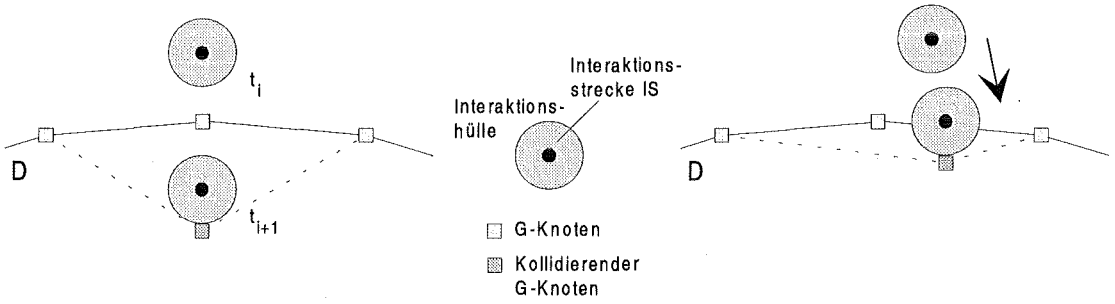


Bild 4.5: Kollision bei zeitlicher Diskretisierung Bild 4.6: Kollision bei räumlicher Diskretisierung

Für diese Plausibilitätsprüfung müssen weitergehende Informationen über das deformierbare Objekt sowie die Funktionalität des interagierenden Instrumententeils herangezogen werden. Eine wichtiger Faktor ist der Oberflächennormalenvektor des Objektes an der Stelle des betroffenen G-Knotens.

Berechnung der Oberflächennormalen

Aus den Informationen, die aus dem MESD-Modell abgeleitet werden können, ist der Normalenvektor des G-Knotens zu bestimmen. Dieser repräsentiert die geometrische Orientierung der Oberfläche des deformierbaren Objektes. Zur Berechnung können die Positionen der direkten Nachbarknoten als auch die Position des zugehörigen Vaterknotens herangezogen werden. Die Normalenvektoren werden auch zur Berechnung der Gouraud-Schattierung bei der graphischen Repräsentation benötigt (Kapitel 5), sie werden deshalb in den Knotendomänen (Bild 3.27) abgelegt. Bei der Initialisierung wird für jeden Knoten eine verkettete Liste angelegt, in der die Nachbarschaftsbeziehungen (als Zeiger) aufgeführt sind. Diese werden zur Normalenbestimmung herangezogen. Zur Berechnung eignet sich eine regelmäßige Netzstruktur nach Bild (3.15) besonders gut, da hierbei jeder G-Knoten genau vier direkte Nachbarn besitzt.

Der Normalenvektor wird über das normierte Kreuzprodukt der Nachbarknotenvektoren bestimmt:

$$\mathbf{n}_K = ((\mathbf{p}_{NK1} - \mathbf{p}_{NK2}) \times (\mathbf{p}_{NK3} - \mathbf{p}_{NK4}))^\circ \qquad (4.7)$$

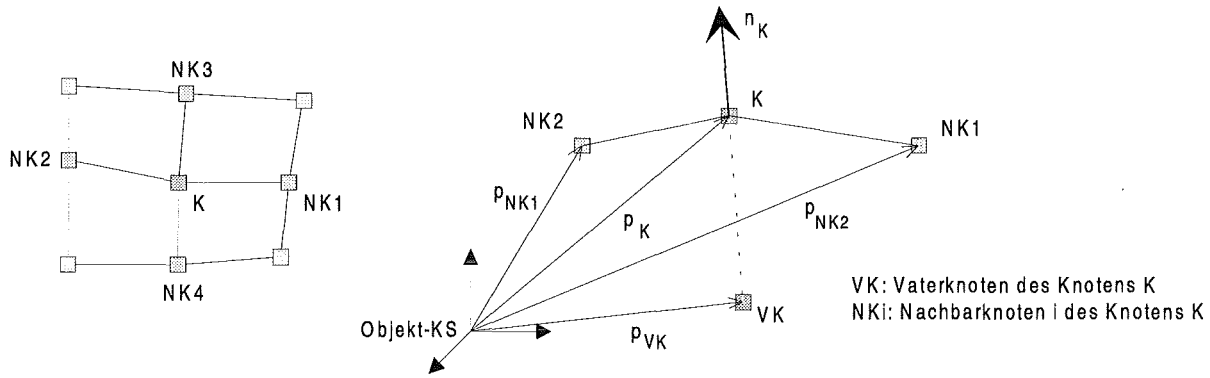


Bild 4.7: Bestimmung des Normalenvektors eines G-Knotens

Alternativ kann bei G-Knoten, die einen Vaterknoten besitzen, der Normalenvektor auch nach Gleichung (4.8) berechnet werden:

$$\mathbf{n}_K = (\mathbf{p}_K - \mathbf{p}_{VK})^\circ \tag{4.8}$$

Diese globale Bestimmung ist zwar schneller, die Oberflächenrichtung wird jedoch weniger gut approximiert, da die lokalen Zustände keinen Einfluß besitzen.

Für jede Interaktionsstrecke wird ein Interaktionswirkungsvektor definiert, der spezifische Eigenschaften des repräsentierten Systems berücksichtigt. Bei Instrumenten ist insbesondere die Definition eines Effektorrichtungsvektors notwendig, der die Wirkungsrichtung des Effektors festlegt. Zur Plausibilitätsprüfung wird der untersuchte Knoten auf die Oberfläche des Ersatz-Instrumentes projiziert (Bild 4.8). Während dieses Projektionsverfahren bei dem (runden) Schaft sehr realitätsnah ist, wird das Interaktionsverhalten des Effektors je nach dessen geometrischer Komplexität mehr oder weniger gut angenähert.

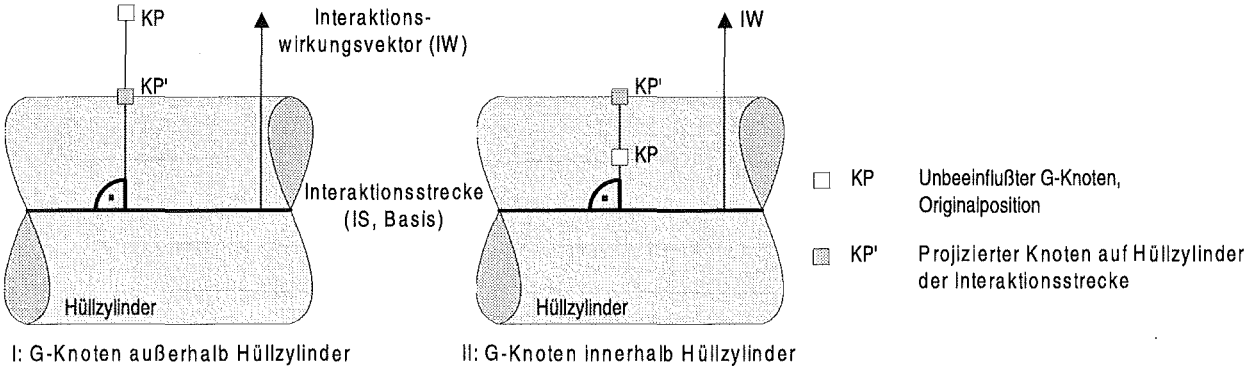


Bild 4.8: Projektion des G-Knoten auf Hüllzylinder der Interaktionsstrecke

Zur Plausibilitätsprüfung werden verschiedene Vektoren des G-Knotens sowie der Interaktionsstrecke miteinander verknüpft (Bild 4.9). Hierzu wird das Skalarprodukt zweier Vektoren mit einer vorgegebenen Schranke verglichen, das Ergebnis ergibt eine binäre Zustandsgröße. Der Plausibilitäts-Schrankenwert kann je nach Instrumententeil und Objekt spezifisch vorgegeben werden, ebenso die zu verknüpfenden Vektorpaare (Tabelle 4.1). Verschiedene Bedingungen werden kombiniert und logisch 'UND'-verknüpft, um eine möglichst gute Aussage bezüglich der Interaktionswahrscheinlichkeit treffen zu können und damit möglichst alle Interaktionsmöglichkeiten zu erfassen. Sind alle einzelnen Prüfungen erfolgreich, so ist auch der Plausibilitätstest im ganzen erfolgreich - eine Interaktion ist

'sinnvoll'. Der normierte Wahrscheinlichkeitswert pb_val bezüglich Richtungsübereinstimmung der Vektoren **v1** und **v2** wird definiert durch:

$$\text{pb\_val} = \left(\frac{\mathbf{v1}}{\|\mathbf{v1}\|} \cdot \frac{\mathbf{v2}}{\|\mathbf{v2}\|} + 1.0\right) \cdot 0.5 \quad , \qquad \text{pb\_val} \in [0,1] \tag{4.9}$$

Die Boolsche Variable plaus_bool repräsentiert das Prüfungsergebnis eines Einzeltests, wobei der Schwellwert der Plausibilitätsprüfung *thr_val* spezifisch festlegbar ist:

$$\text{plaus\_bool}_i = \begin{cases} \text{TRUE} & \textit{für} \ \ \text{pb\_val}_i \geq thr\_val_i \\ \text{FALSE} & \textit{für} \ \ \text{pb\_val}_i < thr\_val_i \end{cases} \tag{4.10}$$

$$\text{plaus\_bool} = \text{plaus\_bool}_1 \wedge \text{plaus\_bool}_2 \wedge \cdots \wedge \text{plaus\_bool}_n \tag{4.11}$$

Weiterhin ist neben der logischen Verknüpfung der n Einzelbedingungen auch eine kombinierte Prüfung möglich. Hierzu werden alle Bedingungen gemeinsam ausgewertet (Produkt aller n Wahrscheinlichkeitswerte ⇔ Produkt aller zugehörigen n Schwellwerte).

$$\text{pb\_val} = \prod_{i=1}^{n} \text{pb\_val}_i \ , \qquad thr\_val = \prod_{i=1}^{n} thr\_val_i \tag{4.12}$$

Das Verfahren der Plausibilitätsprüfung kann auch als 'unscharf' bezeichnet werden[5], da im Gegensatz zu den geometrischen Vorprüfungen keine festgelegten Kollisionsbereiche existieren. Vielmehr fließt spezifisches 'Wissen' über die Art der Interaktion (Instrumententeil, lokaler Objektzustand) und weitere Nebenbedingungen in die Entscheidung ein.

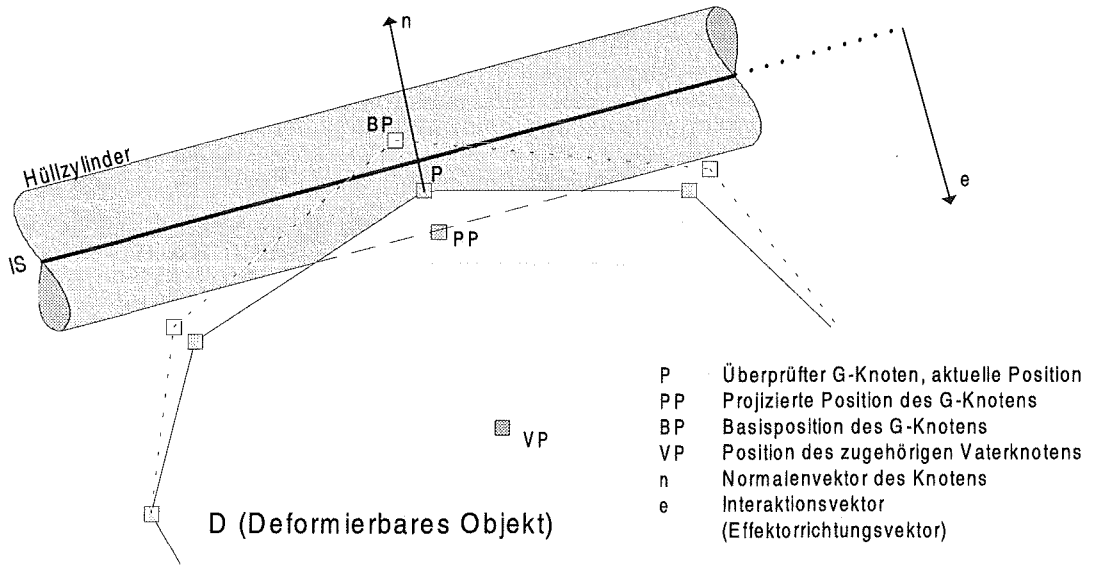


Bild 4.9: Vektoranordnung der Plausibilitätsprüfung

Tabelle 4.1: Auswahl der Vektorpaare der Plausibilitätsprüfung

v1 = n v2 = PB - PP	Genereller Oberflächentest: Eindringung nur nach Innen
v1 = n v2 = PP - VP	Erweiterter Oberflächentest: Kontakt von Innen
v1 = - n v2 = e	Interaktionswirkungstest (Effektorwirkungsrichtung)
v1 = e v2 = PP - P	Interaktionsrichtungstest
v1 = n v2 = P - PP	Positionswirkungstest
v1 = e v2 = P - PP	Inverser Interaktionsrichtungstest

[5] Eine Analogie zu 'Fuzzy'-Systemen in der Regelungstechnik ist durchaus gegeben. Ergebnis ist hier jedoch keine kontinuierliche Regelgröße, sondern Aussagen über den Interaktionszustand.

Die Implementierung der Plausibilitätprüfung ist in Anhang A näher spezifiziert. In Tabelle A.4 ist die Festlegung der herangezogenen Testvektoren erläutert. Die Plausibilitätsprüfung kann für die Interaktionsstrecken und damit für die jeweiligen Instrumententeile verschieden definiert werden, die Testvektoren und Schwellwerte können somit dem spezifischen Instrument angepaßt werden. Die zur Definition mit einem Skript-Kommando benötigten Token sind in Tabelle A.5 aufgeführt.

4.3 Interaktionsauswertung

Wurde für einen Knoten eine Kollision detektiert, so muß eine entsprechende Reaktion des Objektes auf die Anregung erfolgen. Dies erfordert eine spezifische Auswertung der Interaktion und insbesondere eine Manipulation des physikalischen Modells.

4.3.1 Klassen der Manipulation

Bemerkung 4.3:

Die **Manipulation** deformierbarer Körper umfaßt zwei prinzipiell verschiedene Möglichkeiten:

- eine reine morphologische Änderung, also eine Stimulierung der Elastodynamik mit Einwirkung auf die gesamte Bewegungsdynamik. Im wesentlichen werden hier die G- und Z-Vektoren des MESD-Systems beeinflußt. Diese Art der Manipulation wird hier als **Modellverformung** bezeichnet.
- eine strukturelle Modellwandlung durch Neudefinition des deformierbaren Objektes (Änderung der Anzahl der Knoten oder Verbindungselemente, Topologie, Parameter). Dies wird als **Modellmodifikation** bezeichnet.

Die Reaktion und das Verhalten des simulierten Objektes bei Einwirkung eines Instrumentes soll möglichst realistisch auf das physikalische Modell übertragen werden. Im wesentlichen können drei Manipulationseinwirkungen unterschieden werden:

- 'stumpfer' Kontakt, also eine rein verformende Kollision ohne Modellmodifikation.
- 'scharfer' Kontakt, die Kollision bewirkt eine Modellmodifikation. Der scharfe Kontakt wird durch einen entsprechenden Effektortyp ('Messer') oder einer hohen Kraft ('Reißen') ausgelöst.
- Effektorbetätigung, bewirkt eine Aktivierung der Instrumentenfunktionalität. Je nach Effektortyp kann dies nur eine Modellverformung (Faßzange) oder auch eine Modellmodifkation (Schere) zur Folge haben.

Der stumpfe Kontakt kann mit dem auf physikalischen Gesetzmäßigkeiten basierenden MESD-Modell behandelt werden. Bei der Simulation des scharfen Kontakts und der Effektorwirkung hingegen muß auch auf ein Verhaltensmodell und Interaktionsregeln zurückgegriffen werden, um realistische Manipulationen zu ermöglichen. Die Auswertung jeder erfaßten Interaktion erfolgt somit zweistufig: für jeden kollidierenden G-Knoten wird der stumpfe Kontakt betrachtet und die entsprechende Reaktion des Knotens bestimmt. Sind die Voraussetzungen für eine über die Modellverformung hinausgehende Einwirkung erfüllt, so erfolgt eine weitergehende Auswertung der Manipulation. Diese wird in Kapitel 4.4 betrachtet.

4.3.2 Interaktionsattribute und Interaktionsstrukturen

Die Kollisionserkennung und hierbei insbesondere die Plausibilitätsprüfung liefern schon wichtige Informationen über die Art der Manipulation. Alle kollidierenden G-Knoten werden in eine objektspezifische Interaktionsliste eingereiht. Dieses Vorgehen besitzt den Vorteil, daß die schon interagierenden Knoten nach dem nächsten Zeitschritt explizit ausgewertet werden können, ohne den gesamten Zyklus der Kollisionserkennung zu durchlaufen. Der Plausibilitätstest kann für diese Knoten modifiziert und vereinfacht werden. Außerdem ist es hiermit möglich, mit jedem Knoten Vergangenheitsinformationen zu verwerten, um die entsprechenden Interaktionsauswertungen auszuführen. Besonders effizient ist diese Maßnahme, um unerwünschte Objekt-'Durchdringungen' aufgrund der Diskretisierungen zu verhindern. Ein Beispiel soll dies verdeutlichen: Hat ein G-Knoten mit der Außenseite des Effektors interagiert, so ist nach dem nächsten Zeitschritt eine Interaktion mit der Effektorinnenseite nicht möglich.

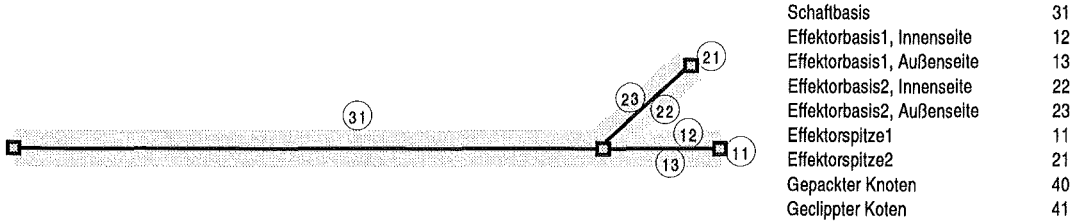


Bild 4.10: Zuweisung der Interaktionsattribute

Jeder interagierende G-Knoten erhält ein Interaktionsattribut, das den kontaktierten Instrumententeil kennzeichnet (Bild 4.10). Ist ein G-Knoten schon in der Interaktionsliste vorhanden, so kann für ein realistisches und plausibles Interaktionsverhalten ein Regelsatz aufgestellt werden (0 - keine Interaktion):

Aktuelles Interaktionsattribut	0	31	12	22	13	23	11	21
Mögliche Interaktionen	Alle	0, 13, 23	0, 22, 11	0, 12, 21	0, 11, 23	0, 21, 13	0, 12, 13	0, 22, 23

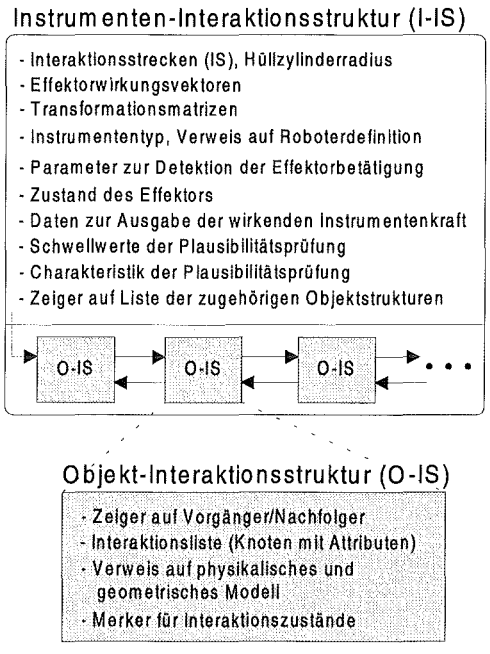


Bild 4.11: Schema der Interaktionsstruktur

Desweiteren wird oftmals die Bestimmung des am Instrument wirkenden effektiven Kraftvektors benötigt, beispielsweise für ein System zur Kraftrückkopplung. Hierzu werden die resultierenden Kräfte der Knoten in der Interaktionsliste aufsummiert, da diese Knoten eine Kraft auf das Instrument ausüben. Die Kräfte werden entsprechend gewichtet, um einen gemeinsamen, an der Effektorspitze angreifenden Kraftvektor abzuleiten.

Bezüglich der Implementierung ist die Einordnung der Interaktionsauswertung in den Berechnungsablauf des physikalischen Modells wichtig (Bild 4.12, zum Vergleich Bild 3.19). Der Interaktionsblock wird pro Zeitschritt für alle definierten Instrumente durch-

laufen, er schließt sich direkt an die Berechnung der Knotendynamik an. Bei k Instrumenten und n deformierbaren Objekten sind somit (n • k) verschiedene Interaktionspaare möglich. Schon zur Initialisierung des Szenarios werden für jedes Instrument spezifische Interaktionsstrukturen definiert. Diese erlauben eine effiziente Organisation der Modellmanipulationen, um den Datenverwaltungsaufwand zur Simulationslaufzeit zu minimieren (Bild 4.11).

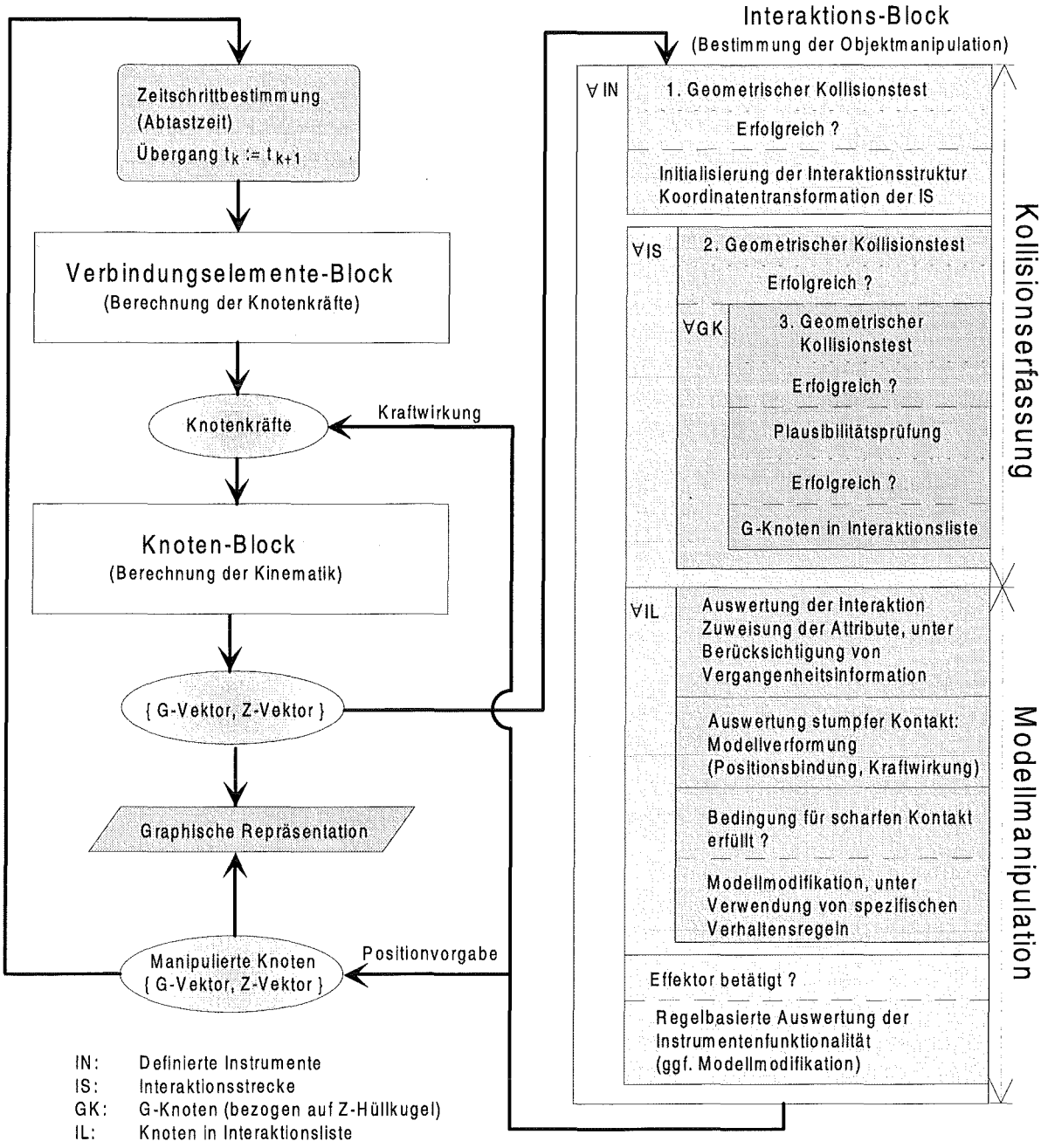


Bild 4.12: Schematischer Ablauf der Interaktionsbehandlung[6]

[6] Vereinfachte Darstellung des Struktogramms, das die Interaktionschritte grob verdeutlichen soll und deshalb nur unvollständig ist. Bei negativem Test erfolgt Abbruch der jeweils inneren Schleife.

4.3.3 Stumpfer Kontakt: Modellverformung

Eine verformende Objektmanipulation bedingt das Einwirken in die Dynamik des physikalischen Modells. Dies kann im Rahmen des MESD-Verfahrens auf zwei Arten geschehen:

- **Positionsvorgabe**:
  Hierbei erfolgt ein direkter Einfluß auf den G-Vektor des interagierenden Knotens. Die aktuelle Knotenposition wird auf den Hüllzylinder der betreffenden Interaktionsstrecke projiziert (Bild 4.8) und die neue Position auf den G-Vektor übertragen. Zusätzlich kann auch eine Einschränkung der Freiheitsgrade über die Beschränkungsmatrix **B** erfolgen (Kapitel 3.3.2).

- **Kraftwirkung**:
  Ein Eingriff in die Modelldynamik kann auch durch Berücksichtigung einer weiteren Kraft zu dem Knotenkraftvektor $\boldsymbol{f}$ erfolgen (Gleichung (3.72)), der als Eingangsgröße des Berechnungsalgorithmus nach Gleichung (3.70) dient. Diese Kollisionskraft $\mathbf{F_{Kol}}$ charakterisiert die vom Instrument ausgeübte Wirkung auf den interagierenden Knoten. Zur Vermeidung von Instabilitäten sollte die Kollisionskraftfunktion stetig sein, man kann sich ein kontinuierliches Kraftfeld nach Bild 4.13 vorstellen. Die Richtung der Kollisionskraft zeigt stets senkrecht von der Interaktionsstrecke IS weg, $\mathbf{F_{Kol}}$ ist eine hyperbolische Funktion des Knotenabstandes von IS nach Gleichung (4.12).

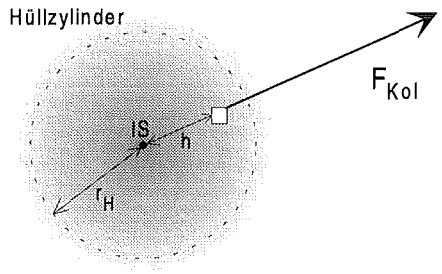


Bild 4.13: Kraftfeld der Kollision

$$F_{Kol} = k_{Kol} \cdot \frac{r_H}{h} \qquad (4.13)$$

h: Abstand des Knotens von der Interaktionsstrecke
r_H: Hüllzylinderradius
k_{Kol}: Kollisionskraftfaktor

Bei der Implementierung hat sich bei Kollisionen von G-Knoten der Ansatz der Positionsvorgabe als geeigneter herausgestellt, da er weniger Stabilitätsprobleme verursacht. Für realistische Interaktionen muß der Verstärkungsfaktor beim Kraftwirkungs-Ansatz sehr hoch[7] gesetzt werden, hieraus entstehen Schwingungen bei der Kollision ('steifes' System). Eine Freiheitsgradbeschränkung findet bei stumpfen Kontakten nicht statt. Die zugehörigen Vaterknoten jedes interagierenden G-Knotens werden auch auf Kollisionen untersucht, aufgrund der innenliegenden und gedämpfteren Anordnung können diese effizient mit der Methode der Kraftwirkung behandelt werden.

Bisher wurden nur Kollisionen von deformierbaren Objekten mit Instrumenten behandelt, die durch Interaktionsstrecken repräsentiert werden. Ein weitere Möglichkeit ist jedoch auch die Kollision zweier deformierbarer Objekte. Die Behandlung dieses Problems ist unter Echtzeitbedingungen ungleich schwieriger, da die Objekte nicht durch Interaktionsstrecken abstrahiert werden können. Durch Definition eines Kraftwirkungsfelds um die Z-Knoten eines Objektes wird es möglich, diesen Fall im Simulationsszenario zu berücksichtigen - wenn auch nur stark vereinfacht. Eine andere Möglichkeit besteht im Einsatz von beschränkten globalen Verbindungselementen zwischen den Knoten benachbarter Objekte (Kapitel 3.2.4). Obwohl hierdurch keine exakten Kollisionsdetektionen vorgenommen und Eindringungen verhindert werden können, ist dieses Vorgehen im Rahmen der Aufgabenstellung ausreichend.

[7] Die Kollisionskraft muß um etwa zwei Größenordnungen stärker sein als die wirkenden VE-Kräfte, ansonsten erfolgt eine Eindringung in die Instrumentengeometrie.

4.4 Modellmodifikation

Der modifizierende Eingriff an einem Objekt erfordert die Neudefinition des MESD-Modells. Allein mit physikalischen Gesetzmäßigkeiten[8] läßt sich die Reaktion nicht definieren, es bedarf weiteren Wissens über das reale Verhalten des Objektes. Dieses Wissen wird in Form von Regeln spezifiziert, um die entsprechenden Auswirkungen auf das Modell zu beschreiben.

4.4.1 Verhaltensmodell

Das Verhaltensmodell bestimmt die Organisation und Durchführung der Modellmodifikation. Es besteht aus einem Satz von Regeln, die das reale Objektverhalten möglichst gut abbilden. Hierin werden die Maßnahmen definiert, die für den Ablauf der Modellmodifikation notwendig sind (Simulationsschritte). Die Verhaltensregeln werden aus praktischen Kenntnissen abgeleitet und aufbereitet, damit die Simulation darauf zurückgreifen kann. Ein Verhaltensmodell kann daher auch nie vollständig sein, sondern repräsentiert nur den Ausschnitt der Realität, der für die jeweilige Anwendung relevant ist.

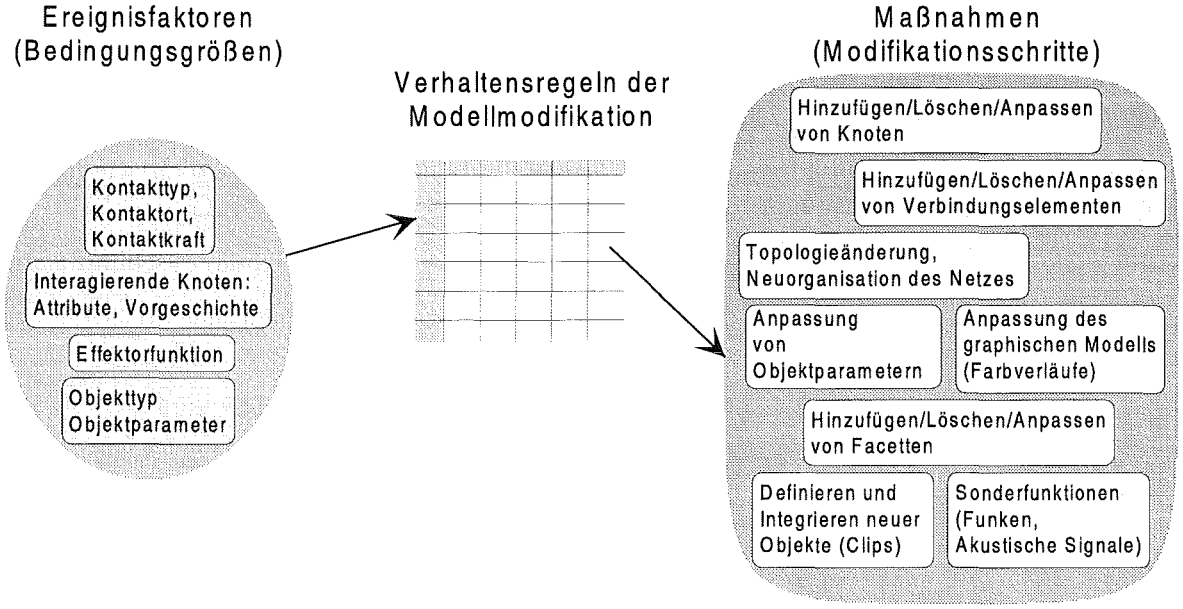


Bild 4.14: Verhaltensmodell bei Modifikationen

Ein Beispiel aus der Minimal-Invasiven Chirurgie ist das (stumpfe) Präparieren, also das Trennen von Organstrukturen durch mechanische Einwirkung des Instrumentes ohne Verwendung der Effektorfunktionalität. Ein einfaches Verhaltensmodell basiert auf dem Vergleich der ausgewirkten Kraft auf einen G-Knoten mit einer Kraftschwelle. Das Überschreiten dieser definierten Zerreißkraft löst ein regelbasiertes Vorgehen aus, das als Resultat ein realistisches Reißen simulieren soll. In obiger Definition wird diese Manipulation als 'scharfer Kontakt' bezeichnet.

Die Berücksichtigung der Instrumentenfunktionalität beruht ebenso auf einem vom Effektortyp abhängigen Verhaltensmodell. Die Auswertung erfolgt hier über eine binäre Funktion, die bei Überschreitung eines Effektorschwellwertes die entsprechende Reaktion des Modells auslöst. In Kapitel 4.5 werden verschiedene Effektorfunktionalitäten aufgezeigt.

[8] Zumindest nicht mit einfachen physikalischen Gesetzen, die innerhalb einer Echtzeitsimulation berücksichtigbar sind.

In der Implementierung dieser Arbeit sind die Regeln in Form von bedingten Verzweigungsketten realisiert. Wichtige Modifikationsparameter sind innerhalb der Objektdefinition festlegbar (Anhang B). Viele Modellmodifikationen lassen sich auf eine Auftrennung des nodalen Netzes zurückführen, die deshalb im folgenden näher betrachtet werden soll.

4.4.2 Netztrennung

Das Auftrennen des nodalen Systems ist beispielsweise bei der Simulation von Schneidevorgängen notwendig. Dies bedingt einen Eingriff in die Struktur des nodalen Netzes mit entsprechender Umorganisation der Netztopologie sowie der Parameteranpassung von Knoten und Verbindungselementen.

Die interne Datenrepräsentation ist hierbei von entscheidender Bedeutung. Das MESD-Verfahren ermöglicht effiziente Modellmodifikationen. Die entsprechenden Datenstrukturen wurden hierzu optimiert, damit Knoten und Verbindungselemente einfach und schnell eingefügt oder gelöscht werden können (doppelt verzeigerte Liste, Bild 3.27 in Kapitel 3.4.2).

Zur Steigerung der Effizienz zur Simulationslaufzeit findet eine möglichst umfangreiche Vorverarbeitung zum Initialisierungszeitpunkt statt. Für jeden Knoten werden daher spezifische Listen angelegt, die wichtige Modifikationsinformationen bereithalten. Das Durchsuchen des gesamten Datensatzes eines Objekts nach betreffenden Komponenten ist daher nicht notwendig.

- *Nachbarschaftsliste*: enthält Zeiger auf alle in räumlichem Bezug stehenden Knoten mit entsprechenden Lageattributen
- *VE-Liste*: enthält Zeiger auf alle Verbindungselemente, die mit dem Knoten verknüpft sind
- *Facettenliste*: enthält Zeiger auf die Facetten, die dem G-Knoten zugeordnet sind

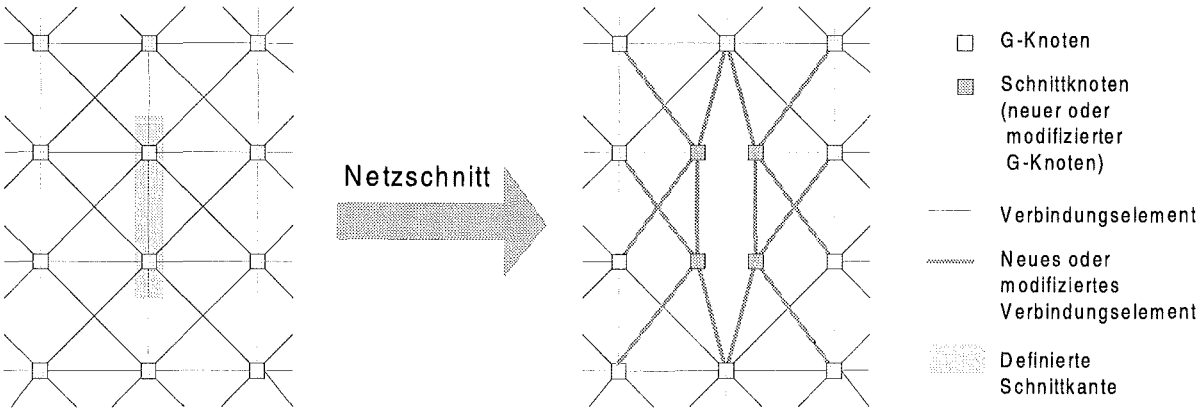


Bild 4.15: Lokales Schnittverfahren bei Oberflächennetz

Folgende Schritte werden unter Vorgabe der Verhaltensregeln zur Netztrennung abgearbeitet:

- Definition der Schnittkante, also Lokalisation der Schnittlage je nach Effektor, Manipulationsregel sowie eventuell schon vorhandener Netztrennung (Mehrfachschnitt)
- Erfassung der betreffenden Knoten ('Schnittknoten') und Filterung, um unstetige Schnittkanten zu vermeiden, Einbeziehung der entsprechenden Vaterknoten

- Für alle Schnittknoten: Generierung eines 'Partnerknotens' mit gleichen Attributen und Parametern und Einordnung in die Datenstruktur, Anpassung der Knotenbeziehungen (Vaterknoten)
- Auftrennung der Verbindungselemente entlang der Schnittkante, Umorganisation oder Löschen aus der Datenstruktur
- Einfügen neuer Verbindungselemente zwischen den Schnittknoten, Initialisierung der VE-Parameter nach spezifischer Vorgabe des Verhaltensmodells (Modifikationsparameter)
- Zur Sicherung der Datenkonsistenz müssen nach jeder Modellmodifikation die Knotenlisten für alle betroffenen Komponenten aktualisiert werden (Nachbarschaftsliste, VE-Liste, Facettenliste)

Ein einfaches Beispiel für das Auftrennen von Netzen zeigt Bild 4.15. Das Durchtrennen von Objekten erfordert auch die Entkopplung der zugehörigen Vaterknoten, wie Bild 4.16 veranschaulicht.

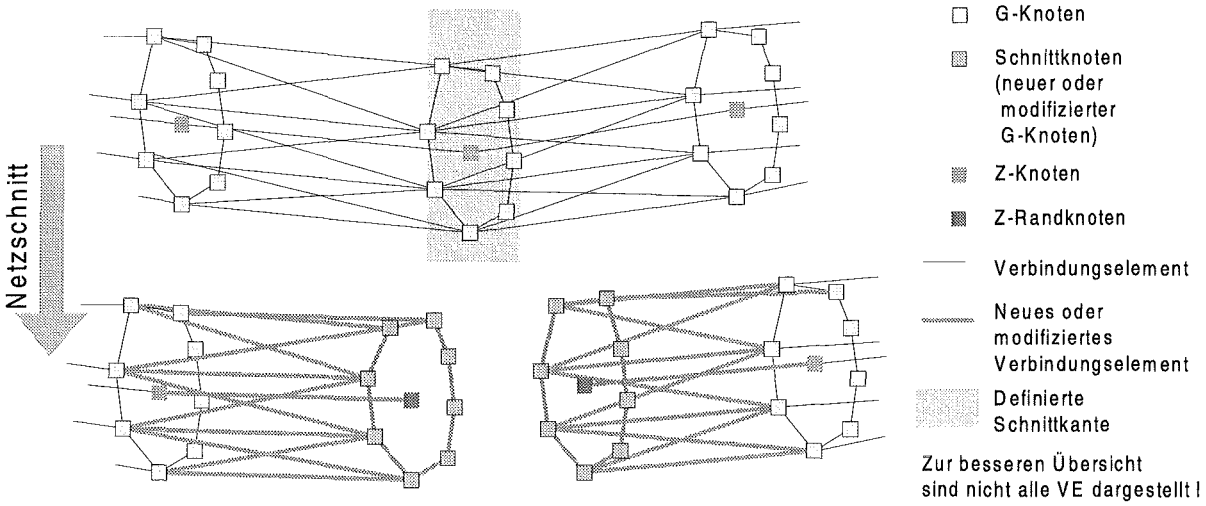


Bild 4.16: Schnittverfahren bei Durchtrennen

Hinsichtlich der graphischen Ausgabe kann eine Erweiterung des geometrischen Modells und damit der Facettenstruktur (Bild 5.5) erforderlich sein. Da die geometrische Modellbildung auf Oberflächenmethoden basiert, ist nach einem Trennvorgang je nach Ansicht die Rückseite der Facetten zu erkennen. Bei manchen Objekten (Gefäßen) kann dies erwünscht sein, bei anderen hingegen ist das Einfügen von Schnittfacetten notwendig, um weiterhin eine geschlossene Geometrie zu erhalten.

4.5 Implementierung chirurgischer Effektorfunktionalitäten

Für die Anwendung als Simulationsumgebung für die Minimal-Invasive Chirurgie ist die originalgetreue Simulation chirurgischer Tätigkeiten notwendig. Innerhalb der vorliegenden Arbeit wurden vier verschiedene Instrumententypen implementiert[9] und deren Funktionalität nachgebildet. Die grundlegenden operativen Eingriffe in der endoskopischen Chirurgie sind damit abgedeckt. Mit den in diesem Kapitel vorgestellten Methoden wurde die Basis zur Simulation weiterer Effektorfunktionalitäten geschaffen.

[9] Die Geometrie von Instrumenten und Effektoren ist mit herkömmlichen Methoden der CAD-Technik modellierbar und kann bei gleicher Funktionalität auch verschiedene Ausprägungen besitzen.

4.5.1 Greifen

Mit einem **Greifer** oder einer **Faßzange** wird das Fassen von Gewebe oder Organstrukturen ermöglicht. Innerhalb der Simulation werden hierzu ein oder mehrere Knoten des deformierbaren Objektes gepackt und fest an den Effektor gebunden, neben der Positionsvorgabe erfolgt also auch eine Beschränkung der Knoten-Freiheitsgrade. Zur Erfassung aller betroffenen Knoten (auch Z-Knoten) wird ein Fangbereich definiert, der die Innenseite des Effektors abdeckt.

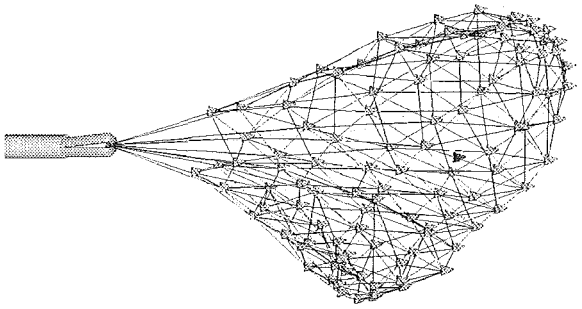


Bild 4.17: Greifen und Ziehen von Gewebe

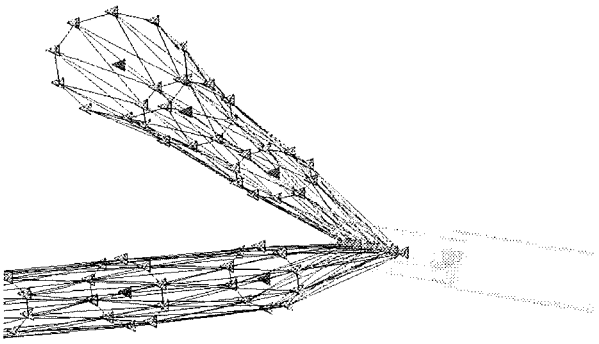


Bild 4.18: Greifen und Ziehen eines Gefäßes

4.5.2 Setzen von Klemmen

Ein **Clip-Applikator** ermöglicht das Setzen von Klemmen (Clips) an Gefäßen oder Gewebe, um einen Flüssigkeitsfluß zu unterbinden und Öffnungen zu verschließen. Hierzu werden analog zum Greifen die vom Clippen betroffenen Knoten erfaßt und dem Clip fest zugeordnet. Weiterhin muß die Geometrie der Klemme definiert und in das Szenario eingebunden werden. Die gesetzte Klemme und alle geclippten Knoten werden einem spezifizierten Z-Knoten (Trägerknoten) zugewiesen und bewegen sich dynamisch mit diesem mit. Die Geometrie der Klemme und die Setzparameter sind über einen Skript-Befehl (Anhang C.2) frei definierbar.

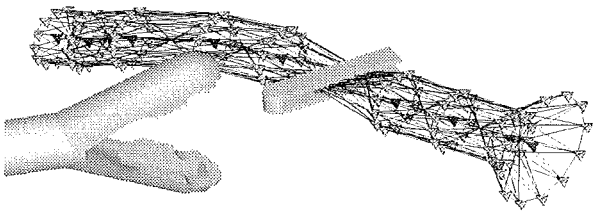


Bild 4.19: Setzen einer Klemme am Gefäß

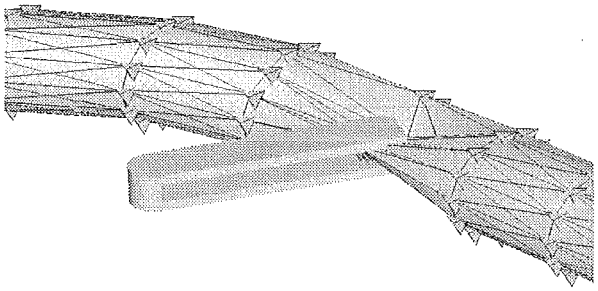


Bild 4.20: Unvollständiges Setzen der Klemme

4.5.3 Schneiden

Eine **Schere** bewirkt das Auftrennen von Oberflächenstrukturen oder auch das Durchtrennen von Gewebe und Gefäßen[10]. In der Simulation ist hierbei eine entsprechende Modellmodifikation notwendig. Durch die Effektorrichtungsvektoren wird die Schnittkante definiert, alle interagierenden Knoten an der 'scharfen' Effektorinnenseite werden zum Trennen herangezogen.

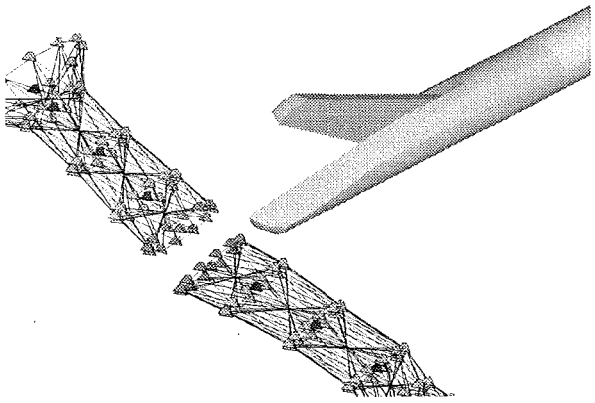


Bild 4.21: Durchschneiden eines Gefäßes

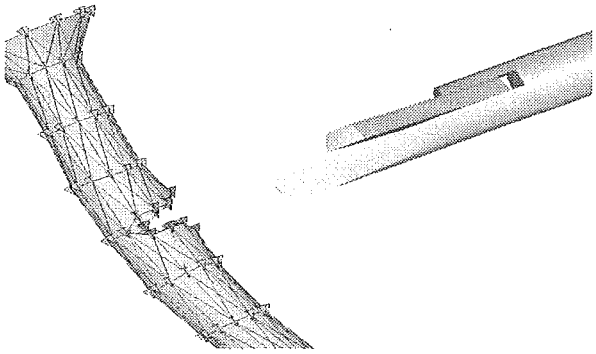


Bild 4.22: Anschneiden eines Gefäßes

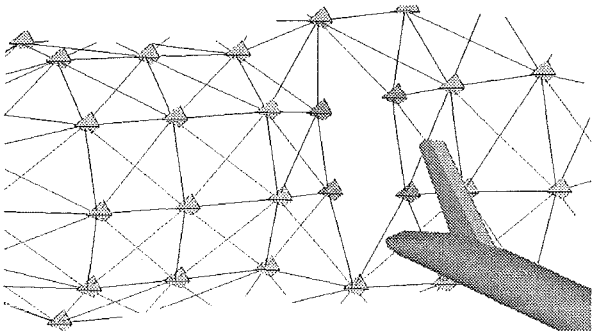


Bild 4.23: Schneiden in Gewebeoberfläche

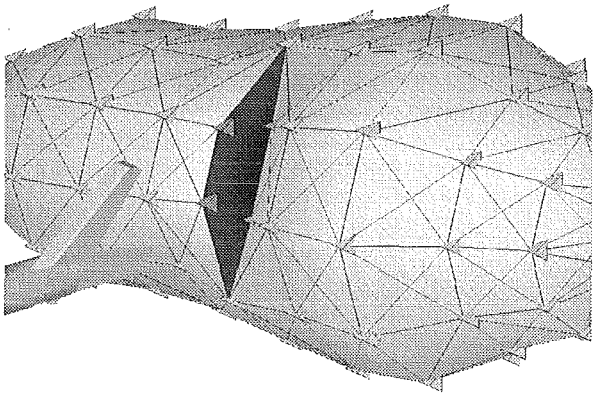


Bild 4.24: wie Bild 4.23, mit graphischem Modell

4.5.4 Koagulieren

Mit Hilfe eines sogenannten **Koagulationshakens** wird ebenfalls das Auftrennen von Oberflächenstrukturen ermöglicht. Hierbei wird durch einen unipolaren Hochfrequenz-Strom eine lokale Erhitzung des Gewebes erzeugt, die eine räumlich beschränkte Verbrennung oder Verödung zur Folge hat. Die Aktivierung erfolgt über einen Fußschalter. Mit einem geringeren Strom kann auch ein Verschweißen von biologischem Gewebe durchgeführt werden. In der Simulation wird das Koagulieren ähnlich wie das Schneiden behandelt. Bei aktivierter Koagulationsfunktion sind alle Effektorteile 'scharf', die Modellmodifikation betrifft somit alle interagierenden Knoten. Zusätzlich wird zur Steigerung des Realismus das graphische Modell beeinflußt, indem die angrenzenden Facetten dunkler eingefärbt werden.

[10] Aus Sicherheitsgründen wird ein Skalpell in der endoskopischen Chirurgie nicht eingesetzt.

Außerdem erfolgt eine Visualisierung von Hochfrequenzeffekten ('Funken'), um den aktivierten Zustand des Instrumentes anzuzeigen.

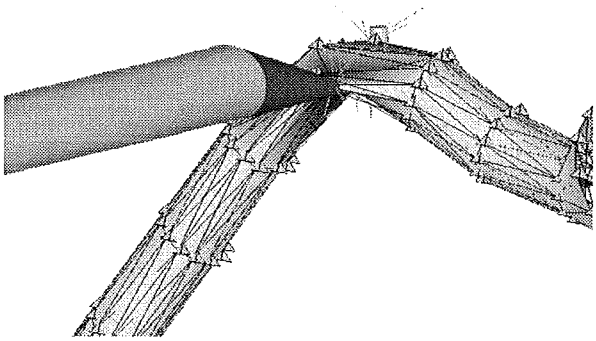


Bild 4.25: Koagulation am Gefäß

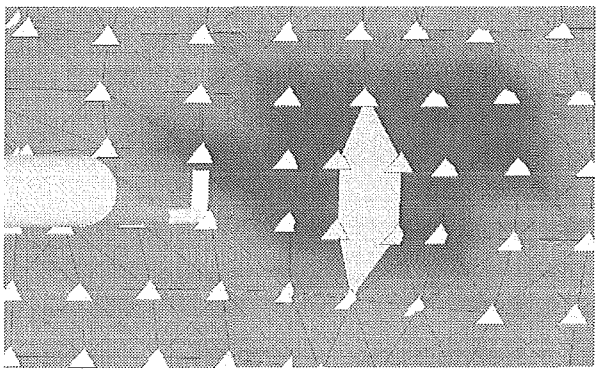


Bild 4.26: Koagulation am Gewebe

4.6 Zusammenfassung des Kapitels

Ein wichtiger Faktor für die Realitätsnähe einer VR-Umgebung ist die Interaktion des Menschen mit dem simulierten Szenario. Im Rahmen dieser Arbeit wird das direkte Eingreifen in die Dynamik und Modellstruktur deformierbarer Objekte behandelt und hierbei insbesondere auf die Anforderungen des MIC-Trainingssystems eingegangen. Auf Basis des MESD-Verfahrens ist eine Behandlung der Interaktion in Echtzeit möglich. Hierzu wurden effiziente Methoden zur Kollisionserkennung, Auswertung von Manipulationen und der Modellmodifikation entwickelt.

Mit Greifen, Clipsetzen, Schneiden und Koagulieren wurden neben dem stumpfen Präparieren die wichtigsten Tätigkeiten der endoskopischen Chirurgie implementiert. Diese Funktionalität bildet die Grundlage für ein realistisches und effektives Training.

5 Graphische Repräsentation deformierbarer Objekte

In der vorgestellten Simulationsumgebung erfolgt eine visuelle Rückkopplung, also eine graphische Darstellung der deformierbaren Objekte auf einem entsprechenden Ausgabesystem. In diesem Kapitel sollen die hierfür nötigen graphisch-geometrischen Grundlagen eingeführt werden. Im 'Stand der Technik' wurden schon verschiedene Möglichkeiten der graphischen Repräsentation angedeutet. Aus Effizienzgründen werden die deformierbaren Objekte durch reine Oberflächenmodelle abstrahiert, da die gängige Graphikhardware auf diese Darstellungsform optimiert ist. In dieser Arbeit werden zwei Ansätze betrachtet: Freiformflächen und reine polygonale Darstellung. Durch die generelle Trennung von physikalischer Repräsentation und graphischer Darstellung ist die Auswahl optional, die Darstellung kann also stets zweigleisig erfolgen.

5.1 Freiformflächen (NURBS)

5.1.1 Grundlagen

Freiformflächen sind gekennzeichnet durch eine mathematisch geschlossene und exakte Beschreibung der Oberflächen mittels algebraischer Gleichungen. In der häufig verwendeten parametrischen Form ist eine Fläche S im dreidimensionalen Raum definiert durch die Parameterfunktion [HL89]:

$$S(s,t) = (x(s,t), y(s,t), z(s,t)), \quad \text{wobei} \quad s \in [s_{min}, s_{max}], \quad t \in [t_{min}, t_{max}] \tag{5.1}$$

x, y, und z sind hierbei beliebige Funktionen der Flächenparameter s und t. Äquivalent kann eine Kurve C im dreidimensionalen Raum parametrisch definiert werden. Das Laufintervall des Kurvenparameter s spezifiziert hierbei das resultierende Teilstück der Kurve.

$$C(s) = (x(s), y(s), z(s)), \qquad \text{wobei} \quad s \in [s_{min}, s_{max}] \tag{5.2}$$

Ein gebräuchlicher Spezialfall ist die Beschränkung auf stückweise polynomiale Parameterfunktionen der Form

$$x_i(s) = a_{0,i} + a_{1,i} \cdot s + a_{2,i} \cdot s^2 + \ldots + a_{n,i} \cdot s^n \qquad s \in [s_i, s_{i+1}] \tag{5.3}$$

$$s_{min} = s_0 \leq s_1 \leq \ldots \leq s_i \leq s_{i+1} \leq \ldots \leq s_m = s_{max} \tag{5.4}$$

Eine (n-1)-stetige Kurve, die aus m einzelnen polynomialen Funktionen zusammengesetzt ist, wird ***Spline-Kurve*** des Grades n genannt. Die Parameterpunkte s_i, an denen die polynomialen Kurvenstücke aneinanderstoßen, werden als Knoten des Splines bezeichnet[1]. Um an diesen Knoten stetige Übergänge zu gewährleisten, ist für jede Polynomfunktion eine Berücksichtigung von Randbedingungen notwendig.

Ein B-Spline (Basis-Spline) ist eine besonders kompakte Darstellung der stückweise polynomialen Kurven. Ein B-Spline besitzt einen Satz von charakteristischen Punkten, die ***Kontrollpunkte*** genannt werden. Diese Kontrollpunkte P_i bestimmen mit ihrer Position im Raum die Form der Kurve. Mathematisch beschrieben ergibt sich:

$$C(s) = \sum_{i=0}^{k-1} \mathbf{P}_i \, B_{i,n}(s) \tag{5.5}$$

Die Basisfunktionen $B_{i,n}$ des B-Splines vom Grad n sind rekursiv definiert:

$$B_{i,0}(s) = \begin{cases} 1 & \text{falls} \quad s_i \le s < s_{i+1} \\ 0 & \text{sonst} \end{cases} \tag{5.6}$$

$$B_{i,n}(s) = \frac{s - s_i}{s_{i+n} - s_i} B_{i,n-1}(s) + \frac{s_{i+n+1} - s}{s_{i+n+1} - s_{i+1}} B_{i+1,n-1}(s) \tag{5.7}$$

k: Anzahl der Kontrollpunkte des Splines
n: Grad der Splinefunktion
p: Ordnung der Splinefunktion, p = n + 1
m: Anzahl der Knoten s_i, m = k + p

Die Knoten s_i des Splines können zu einem Vektor, dem Knoten- oder Trägervektor $\mathbf{s}_T = [s_1, s_2, \ldots, s_m]^T$ zusammengefaßt werden, dessen Elemente stets monoton aufsteigend sind. Die Knoten unterteilen die Splinekurve in Segmente S_i. Jedes Kurvensegment ist ein Teilstück der Kurve, das im Parameterintervall $[s_i, s_{i+1}]$ liegt. Die Form und Lage des Segmentes S_i ist nur von den Kontrollpunkten P_i, P_{i+1}, ..., P_{i+p} abhängig, die anderen Kontrollpunkte beeinflussen das Kurvensegment nicht. Umgekehrt kann jeder Kontrollpunkt auf höchstens p Kurvensegmente einwirken. Global betrachtet nähert sich die Kurve den Kontrollpunkten an, durchläuft diese im allgemeinen jedoch nicht (Bild 5.1). Die Stetigkeit in den Übergangspunkten kann durch den Grad der Splinefunktionen vorgegeben werden.

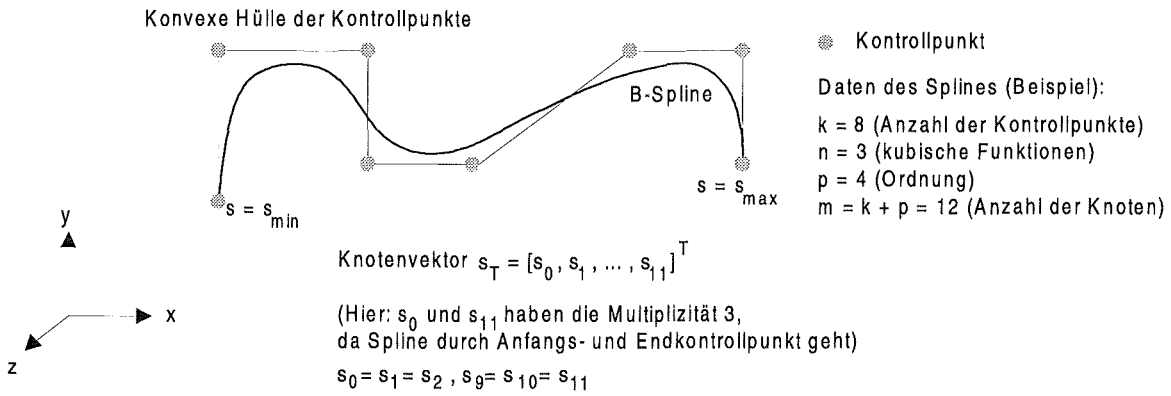


Bild 5.1: Beispiel einer B-Spline-Kurve

[1] Splineknoten sind stets von den Masseknoten der nodalen Netze zu unterscheiden !

Ein Spline mit gleichen Knotenabständen δ wird als ***uniformer Spline***[2] bezeichnet.

$$u_{i+1} = u_i + \delta \quad \text{für} \quad 0 \le i < m-1 \tag{5.8}$$

Analog hierzu wird ein Spline mit wahlfreien Knoten nicht-uniformer Spline genannt, wobei die Knotenfolge weiterhin nicht absteigend sein darf. Dies beinhaltet jedoch auch die Möglichkeit der Definition mehrfacher Knoten mit gleichem Knotenwert. An dieser Stelle verringert sich die Ordnung der Splinefunktion, die Kurve 'schmiegt' sich näher an den korrespondierenden Kontrollpunkt an. Bei der Multiplizität q des Knotens ergibt sich eine Ordnung p'= p - q. Ist q = p - 1, so verläuft die resultierende Kurve durch den Kontrollpunkt, sie ist an dieser Stelle jedoch nur noch C^0-stetig. Will man also erreichen, daß die Kurve direkt am ersten Kontrollpunkt beginnt und am letzten endet, so muß man die Multiplizität des ersten und letzten Knotens auf q = p - 1 festlegen.

Eine Verbesserung der Modelliermöglichkeiten bieten die sogenannten ***rationalen Splines***. Ein Kontrollpunkt $\mathbf{P_{i,h}}$ besitzt hier neben den drei Koordinaten zur Bestimmung der Raumposition im kartesischen Koordinatensystem eine weitere Koordinate, das Gewicht w. Mathematisch bedeutet dies ein Übergang vom affinen Raum in ein homogenes Koordinatensystem [Pie91].

$$\mathbf{P_i} = [x_i \quad y_i \quad z_i]^T \quad \Rightarrow \quad \mathbf{P_{i,h}} = [x_i \quad y_i \quad z_i \quad w_i]^T \tag{5.9}$$

Mit dieser Erweiterung ist ein rationaler B-Spline definiert als das Verhältnis zweier Splines:

$$C(s) = \frac{\sum_{i=0}^{k-1} \mathbf{P_i} B_{i,n}(s)}{\sum_{i=0}^{k-1} w_i B_{i,n}(s)} \tag{5.10}$$

Mit dem Gewicht kann der Einfluß des Kontrollpunktes auf den Kurvenverlauf gesteuert werden. Wird das Gewicht eines Kontrollpunktes erhöht, so schmiegt sich die Kurve näher an den Kontrollpunkt an. Der Einfluß der anderen, nicht modifizierten Kontrollpunkte auf das entsprechende Kurvensegment nimmt gleichzeitig ab. Sind die Gewichte aller Kontrollpunkte gleich 1, so ergibt sich ein nicht-rationaler B-Spline.

Der Übergang von Spline-Kurven auf Spline-Flächen erfolgt durch die Einführung eines weiteren Parameters t, so daß die beiden Flächenparameter s und t ein Rechteckgebiet definieren (Bild 5.2). Über die beiden entsprechenden Splinefunktionen wird das Tensorprodukt gebildet, somit lautet die mathematische Vorschrift zur Darstellung einer rationalen, auf B-Splines basierenden Fläche:

$$S(s,t) = \frac{\sum_{i=0}^{k_s-1} \sum_{j=0}^{k_t-1} \mathbf{P_{i,j}} B_{i,n_s}(s) B_{i,n_t}(t)}{\sum_{i=0}^{k_s-1} \sum_{j=0}^{k_t-1} w_{i,j} B_{i,n_s}(s) B_{i,n_t}(t)} \tag{5.11}$$

Die beiden Splinefunktionen N(s) und N(t) können unterschiedliche Ordnungen und Knotenvektoren besitzen, folglich kann auch die Anzahl der Kontrollpunkte verschieden sein. Die Kontrollpunkte der Splinefläche können in einer Matrix mit der Dimension (k_s x k_t) angeordnet werden, diese Kontrollpunktmatrix **KP** bestimmt die Form und Lage der Fläche.

[2] Eine direkte Übersetzung des englischen Wortes 'uniform' würde 'gleichmäßig, gleichförmig' liefern. Diese Ausdrucksweise ist jedoch ungebräuchlich, 'uniform' findet auch im Deutschen als Fremdwort Verwendung.

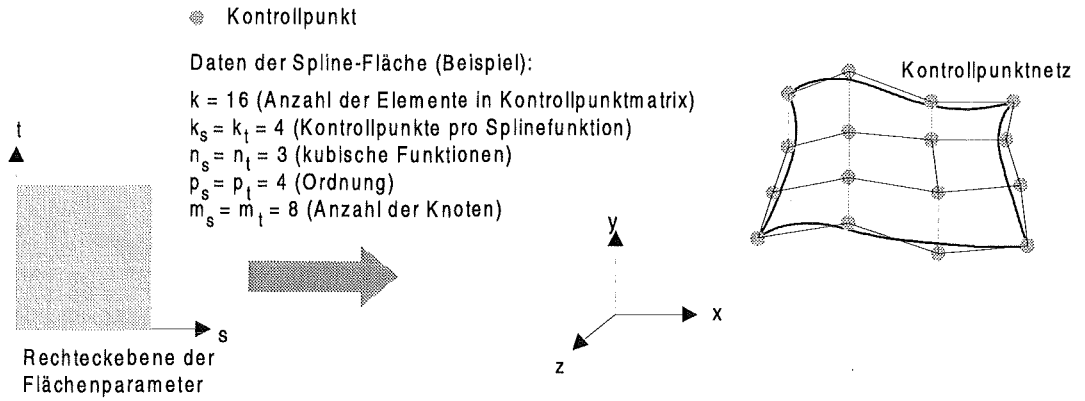


Bild 5.2: Beispiel einer B-Spline-Fläche (NURBS)

Definition 5.1:

Mit **NURBS** wird eine Freiformfläche mit folgenden Eigenschaften bezeichnet:

- **N**on-**U**niform	Die Knoten der Splinefunktionen müssen nicht gleiche Abstände zueinander besitzen, mehrfache Knoten sind möglich
- **R**ational	Definition ist Quotient zweier Splinefunktionen, erlaubt die Verwendung von Gewichten bei Kontrollpunkten
- **B**-Spline	Die Flächen beruhen auf B-Splinefunktionen (Basisfunktionen), die stückweise polynomiale Funktionen repräsentieren
- **S**urface	Parametrische Oberfläche mit den Flächenparametern s und t

Ein NURBS ist vollständig definiert[3] durch eine Kontrollpunktmatrix **KP** sowie zweier Knotenvektoren $\mathbf{s_T}$ und $\mathbf{t_T}$. Zur Auswertung wird die Gleichung (5.11) herangezogen.

Freiformflächen auf Grundlage von NURBS sind in der Computergraphik und bei CAD-Anwendungen sehr beliebt. Dies ist begründet durch die exakte mathematische Darstellbarkeit, die einfache Definition und Auswertung sowie die große Modellierungsfreiheit und klare geometrische Interpretation. Weiterhin sind NURBS invariant gegenüber Skalierung, Rotation, Translation und perspektivischer und paralleler Projektion, eine Eigenschaft, die gerade in VR-Anwendungen sehr wichtig ist. NURBS sind in der Graphikbibliothek GL als Objektprimitiv enthalten. Dadurch ist es möglich, die Graphikhardware einfach und effizient anzusprechen. Für eine weitergehende Betrachtung der NURBS und anderer splinebasierter Flächen wird der Leser auf [HL89], [BBB87], [Far92] oder [Far88] verwiesen.

5.1.2 Anwendung der NURBS für deformierbarer Objekte

Eine zentrale Aufgabe ist die Kopplung des physikalischen Modells mit dem geometrischen Modell, also der NURBS-Darstellung. Die geometriebestimmenden Komponenten des MESD-Modells sind die G-Knoten. Diese müssen auf die Kontrollpunkte übertragen werden, die die Form und Lage des NURBS-Körpers spezifizieren.

[3] Die Ordnung der Splinefunktionen ist durch die Dimensionen der Kontrollpunktmatrix und der Knotenvektoren bestimmt.

Bemerkung 5.1:

Die Kopplung des MESD-Modells mit der Darstellungsform der NURBS erfolgt durch Projektion der Positionskoordinaten der G-Knoten auf die Elemente der Kontrollpunktmatrix **KP**. Hierbei kann ein Objekt auch durch mehrere NURBS repräsentiert werden.

$$\{\text{G} - \text{Vektoren}\} \Rightarrow \mathbf{KP} \tag{5.12}$$

Die G-Knoten dienen hierbei als positionsvorgebende Elemente. Vorteilhaft ist die regelmäßige Anordnung der G-Knoten ebenfalls in Matrixform, wie sie durch die Objektprimitive in Kapitel 3.4.1 vorgegeben ist. Damit ist eine Projektion einfach und effektiv durchführbar. Als zusätzliche Freiheitsgrade der geometrischen Modellbildung können die Trägervektoren und die Wertigkeit der Kontrollpunkte dienen.

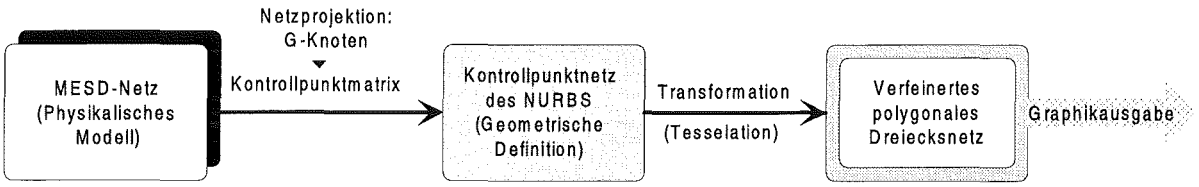


Bild 5.3: Struktur der graphischen Darstellung über NURBS

Die mathematische Definition der NURBS erfolgt in einem relativ hohen Abstraktionsniveau. Zur Ausgabe des Körpers muß deshalb eine Transformation in eine einfachere Darstellungsform durchgeführt werden[4]. Hierzu wird die Oberfläche des NURBS durch ein polygonales Netz approximiert (Tesselation). Durch Anpassung der Tesselationsgenauigkeit kann die Qualität der graphischen Darstellung gesteuert werden, unabhängig vom Umfang des physikalischen Modells (Anzahl der G-Knoten). Weitere Vorteile der NURBS liegen in der einfachen Manipulierbarkeit des Modells durch Verschieben der Kontrollpunkte und in der stets glatten Darstellung des Objektes, welche der Modellierung von deformierbaren Objekten entgegenkommt.

Ein Problem der realitätsgetreuen Modellbildung liegt in dem splinebasierten Charakter der Freiformflächen. Ein Flächensegment wird von mehreren (bei kubischen Funktionen beispielsweise 16) Kontrollpunkten beeinflußt, ein Kontrollpunkt verändert die Form mehrerer Flächensegmente. Das physikalische Modell wird somit durch das geometrische Modell der NURBS überlagert. Außerdem verlaufen die Flächen im allgemeinen nicht durch die Kontrollpunkte, so daß ein Formunterschied zwischen physikalischem und geometrischem Modell existiert. Dies ist insbesondere bei Interaktionen nachteilig. Eine entsprechende Anpassung der Gewichte kann dies ausgleichen. Setzt man das Gewicht eines Kontrollpunktes eine Größenordnung höher als die Gewichte der Nachbarkontrollpunkte, so verläuft die Oberfläche durch diesen Kontrollpunkt, zumindest in visuell ausreichender Näherung.

Ein wichtiger Faktor ist die Darstellbarkeit geschlossener Oberflächen mit NURBS. An den Grenzkurven der jeweiligen NURBS-Flächen - dies können gegenüberliegende Randkurven einer Fläche oder auch verschiedene NURBS sein - müssen hierzu mindestens C^1-stetige Übergänge gewährleistet werden. Diese Randbedingungen sind auf zwei Arten zu realisieren:

[4] Diese Aufgabe wird im allgemeinen vom Graphiksubsystem des Rechners erledigt, eine Anpassung der Genauigkeit ist jedoch möglich.

- Einfügen von Mehrfachknoten am Anfang und Ende des Trägervektors (vgl. Bild 5.1), hiermit lassen sich zwar C^0-stetige Übergänge bilden, C^1-stetige Übergänge erfordern aber weitere, aufwendigere Übergangsbedingungen.
- Wiederholung der ersten p Kontrollpunkte (p: Ordnung des B-Splines) am Ende der jeweiligen Kontrollpunktliste. Aus einer (k_s x k_t) -Kontrollpunktmatrix ergibt sich somit bei einseitig geschlossener Fläche eine erweiterte ((k_s +p_s) x k_t) Kontrollpunktmatrix. Da die Knotenpositionen und damit die Kontrollpunktkoordinaten variant sind, muß die erweiterte Kontrollpunktmatrix vor jeder Graphikausgabe neu generiert werden.

Die Umwandlung der mathematischen Beschreibungsform der NURBS in die geometrischen Daten eines polygonalen Netzes (Tesselation) ist aufwendig und benötigt daher eine relativ große Rechenzeit. Deshalb sind NURBS für die Echtzeitsimulation nur bedingt geeignet[5]. Problematisch ist weiterhin die Möglichkeit der interaktiven Modellmodifikation, also der Strukturveränderung von NURBS bei bestimmten Interaktionen. Durch das hohe Abstraktionsniveau sind viele Übergangs- und Randbedingungen einzuhalten, um die Geschlossenheit und Stetigkeit (C^0/C^1) der Oberflächen zu gewährleisten.

5.2 Polygonale Netze

Bei polygonalen Netzen wird der zu repräsentierende Körper durch ebene Flächenstücke approximiert. Die Gesamtheit dieser zweidimensionalen Facetten im dreidimensionalen Raum formt die geometrische Oberfläche des Körpers. Diese Darstellung wird auch als Polyedermodell bezeichnet. Die Repräsentation mittels polygonaler Netze ist in der echtzeitfähigen Computergraphik und insbesondere bei VR-Anwendungen beliebt, da sie eine sehr schnelle Graphikausgabe erlaubt. Die im Rahmen dieser Arbeit verfügbare Rechnerhardware ist für polygonale Darstellungen optimiert.

5.2.1 Grundlagen und Anwendung bei deformierbaren Objekten

Zur Definition der Geometrie eines Polyedermodells ist für die vorliegende Aufgabenstellung eine Liste aller Facetten des Körpers mit den jeweiligen Koordinaten der Polygoneckpunkte ausreichend. Für das Schattierungsmodell ist zusätzlich die Definition des Normalenvektors für jeden Polygoneckpunkt notwendig. Analog zu der Bemerkung 5.1 läßt sich festhalten:

Bemerkung 5.2:

Die Kopplung des MESD-Modells mit der Darstellungsform polygonaler Netze erfolgt durch Projektion der Positionskoordinaten der G-Knoten auf die Polygoneckpunkte der Facetten. Hierbei wird ein Knoten mehreren ortsgleichen Polygoneckpunkten verschiedener Facetten zugeordnet, da ein definierter Körperpunkt im allgemeinen Eckpunkt mehrerer benachbarter Polygone ist.

Dieses Vorgehen beschleunigt die Grafikausgabe im Vergleich zu NURBS erheblich. Weiterhin ist die geometrische Modellbildung nicht auf regelmäßige Strukturen beschränkt, ebenso sind auch geschlossene Geometrien durch die Projektionsvorschriften erfaßt. Mit Rücksicht auf Interaktionen ist es vorteilhaft, daß die Körperoberfläche durch alle projizierten G-Knoten verläuft. Modellmodifikationen sind einfacher möglich, da Strukturänderungen in einem relativ tiefen Abstraktionsniveau angesetzt werden können.

[5] Diese Aussage wird bei den nächsten Generationen von Graphikrechnern sicherlich keine Gültigkeit mehr besitzen, zumindest wenn die Tesselation von der Graphikhardware übernommen wird.

Nachteilig ist jedoch, daß die Qualität der graphischen Ausgabe direkt vom Modellumfang des physikalischen Modells abhängt (Bild 5.4). Den G-Knoten werden direkt Polygoneckpunkte zugeordnet, eine Gütesteigerung des geometrischen Modells kann nur durch Erhöhung der Anzahl der G-Knoten erfolgen. Im Vergleich zu NURBS verschlechtert sich die visuelle Qualität der Modelle bei gleicher Modellgröße.

In gängigen Graphiksubsystemen ist für Polygone eine (fast) beliebige Anzahl an Eckpunkten möglich, jedoch sollte das Polygon konvex sein und alle Eckpunkte auf einer Ebene liegen [SGI91]. Bei deformierbaren Geometrien ist die Einhaltung dieser Bedingungen für vier oder mehr Polygoneckpunkte nicht möglich. Bei Verwendung von Dreiecksfacetten werden die Anforderungen stets erfüllt, so daß im Rahmen dieser Arbeit nur Polygone mit drei Eckpunkten eingesetzt werden.

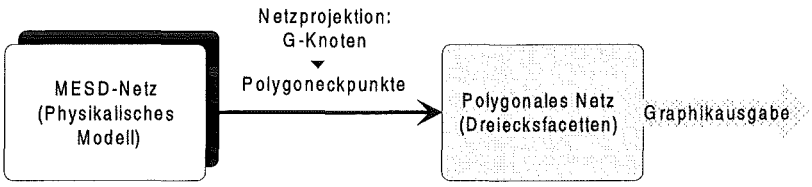


Bild 5.4: Struktur der graphischen Repräsentation über einfache polygonale Netze

Die Repräsentation der Geometriedaten in der rechnerinternen Datenstruktur ist von den Anforderungen abhängig. Innerhalb von KISMET [Kue91] sind die Daten für eine schnellstmögliche Ausgabe aufbereitet, für jede Facette werden die Polygoneckpunkte und die Normalenvektoren in dazugehörigen Listen gehalten, ein Polygonpunkt wird somit mehrfach gespeichert. Dies ist möglich, da die Geometrie und damit die Polygonpunkte invariant sind. Bei deformierbaren Objekten ist diese Vorgehensweise nicht praktikabel, da sich die Koordinaten eines Punktes ändern können und damit die Datenkonsistenz nicht gewährleistet ist. Deshalb wird hier eine facettenbasierte, kompakte Datenstruktur verwendet, bestehend aus einer Liste aller Facetten und einer Liste mit allen Polygoneckpunkten. Letztere entspricht den Knotendomänen von Bild 3.27. Dieser Bereich wird vom Dynamikmodul beschrieben und dann vom Graphikmodul facettenweise gelesen und graphisch dargestellt (Bild 5.5). Ein Eintrag in der Facettenliste repräsentiert eine Facette des Objektes, als wichtigste Komponente besitzt sie drei Verweise auf die jeweiligen Polygoneckpunke (Zeiger). Bei Interaktionen (Modellmodifikationen) kann es notwendig sein, die Facettenliste zu erweitern und zu manipulieren. Hierzu sind die Facetteneinträge auch mit Attributen versehen, die bestimmte Facettentypen charakterisieren (Farbveränderung nach Interaktion, Lochfacetten).

5.2.2 Netzverfeinerung

Um glattere Oberflächen und damit eine realistischere Darstellung zu erreichen, ist eine Verfeinerung des polygonalen Netzes sinnvoll. Trivialerweise kann dies durch eine Vergrößerung des physikalischen Modells und damit eine Erhöhung der Anzahl der G-Knoten erfolgen. Aufgrund der Echtzeitbedingung ist diese Lösung meist unpraktikabel.

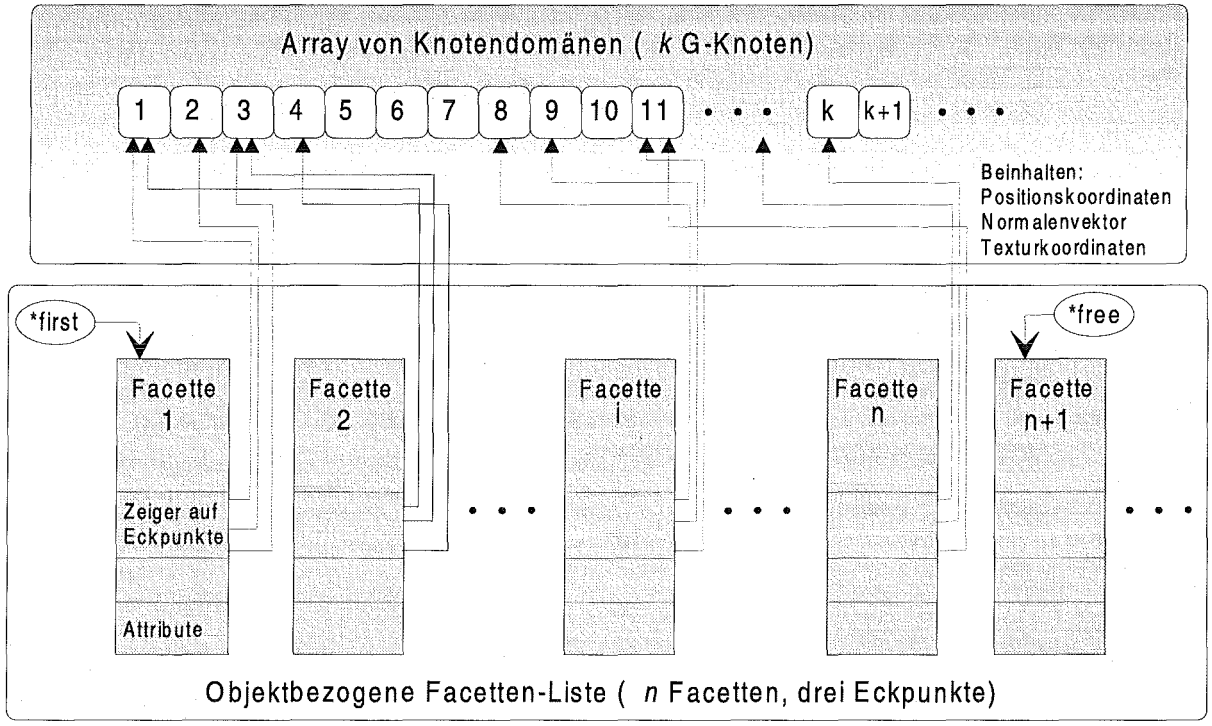


Bild 5.5: Datenstruktur für die polygonale Darstellung deformierbarer Körper

Auch um die Trennung des physikalischen Modells und des geometrischen Modells beizubehalten, ist eine Verfeinerung auf der graphischen Ebene notwendig. Zwischen dem Grundgerüst der vorhandenen Polygoneckpunkte werden weitere geometrische Punkte sinnvoll eingefügt, um mit deren Hilfe ein verfeinertes Dreiecksnetz zu 'spinnen' (Bild 5.6). Hierbei können verschiedene Interpolationsverfahren herangezogen werden, beispielsweise unter Verwendung von Spline-Techniken.

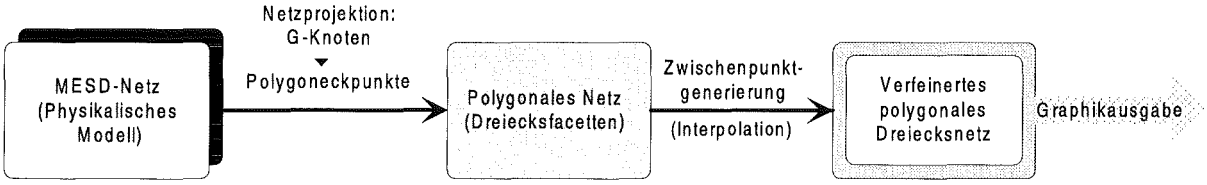


Bild 5.6: Struktur der graphischen Repräsentation über verfeinerte polygonale Netze

Da diese Interpolation bei jedem Darstellungszyklus erfolgen muß, ist ein sehr schnelles und effizientes Verfahren notwendig. Im Rahmen dieser Arbeit wurde eine einfache lineare Zwischenpunktinterpolation implementiert. Die Anzahl der geometriebildenden Polygonpunkte wird hierbei insgesamt verdoppelt, die Facettenanzahl vervierfacht. Aus Effizienzgründen wird keine partielle Verfeinerung durchgeführt, da die Bestimmung von Unstetigkeitskriterien für eine lokale Verfeinerung mehr Rechenzeit benötigen würde als die Interpolation des Zwischenpunktes an sich. Außerdem ist die Wiederherstellung der geometrischen Konsistenz sehr aufwendig, da durch partielle Vereinfachung entstehende Löcher wieder beseitigt werden müssen, um eine geschlossene Geometrie zu erhalten.

Jede Facette der ursprünglichen Geometrie, die ja nur aus dreieckigen Polygonen besteht, wird in 4 Unterfacetten aufgespaltet. Hierzu wird jede Kante des Polygons durch Einfügen

eines Zwischenpunktes in 2 Teilkanten zerlegt. Die Position des neuen Zwischenpunktes wird aus dem Abstand der beiden Kantenpunkte und deren Normalenvektoren bestimmt (Bild 5.7).

Der normierte Normalenvektor **n12** des Zwischenpunktes **P12**, der zwischen den vorhandenen Punkten **P1** und **P2** erzeugt wird, bestimmt sich zu:

$$\mathbf{n12} = (\mathbf{n1} + \mathbf{n2})^\circ = \frac{\mathbf{n1} + \mathbf{n2}}{\|\mathbf{n1} + \mathbf{n2}\|} \tag{5.13}$$

Mit dem Abstandsvektor **p** = **P2** - **P1** ergibt sich für den Interpolationsbasisvektor **a**:

$$\mathbf{a} = k_I(\mathbf{p} \cdot (\mathbf{n1} - \mathbf{n2})^\circ) \cdot \mathbf{n12} \tag{5.14}$$

Hierbei ist k_I eine Interpolationskonstante, die die Glättung in Abhängigkeit der Normalendifferenz bestimmt. Werte um k_I = 0,2 liefern gute Ergebnisse. Für die Koordinaten des Zwischenpunktes **P12** gilt hiermit:

$$\mathbf{P12} = \mathbf{P1} + \frac{\mathbf{p}}{2} + \mathbf{a} \tag{5.15}$$

Die (eventuell nötigen) Texturkoordinaten werden wie der Normalenvektor direkt linear interpoliert.

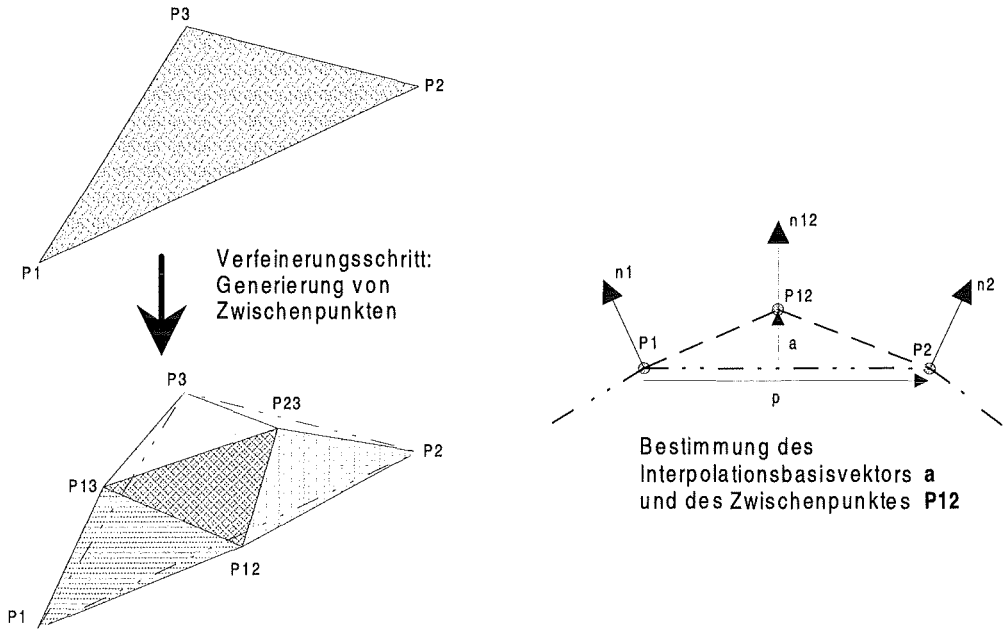
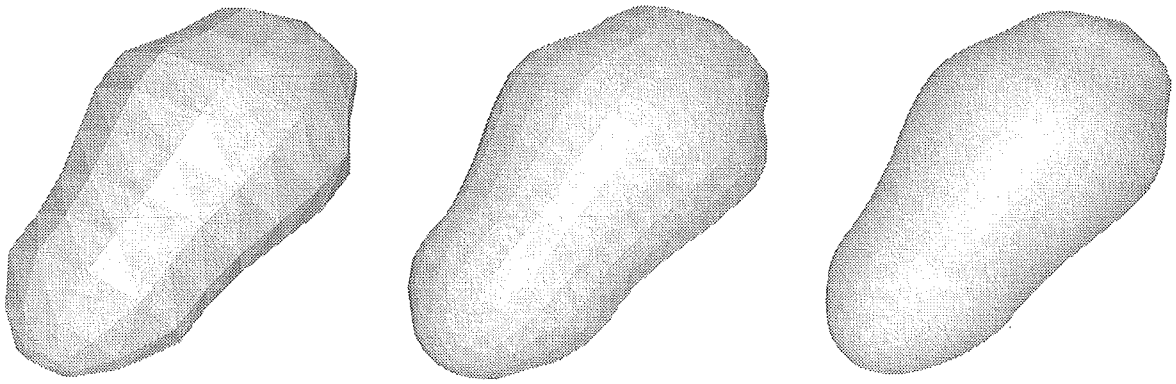


Bild 5.7: Effiziente Verfeinerung polygonaler Netze

Durch die Anwendung dieses Verfahrens entsteht der Eindruck einer glatteren Oberfläche. Die Geometrie des Körpers wird bei dieser Methode nicht besser rekonstruiert, sondern nur die graphische Darstellung 'abgerundet' und erscheint damit hinsichtlich deformierbarer Objekte realistischer. Der Verfeinerungszyklus kann - soweit sinnvoll - beliebig oft wiederholt werden.



Einfacher Polyeder (10,7 ms) Verfeinerter Polyeder (19,7 ms) NURBS (288,8 ms)

Bild 5.8: Darstellungsqualität bei verschiedenen geometrischen Modellen

Bild 5.8 zeigt einen Vergleich der Darstellungsqualität bei verschiedenen graphischen Repräsentationen. Das Modell ist relativ einfach gehalten und nicht schattiert, um die Unterschiede im Bild zu verdeutlichen. Die Zeiten sind grobe Richtwerte[6] zur graphischen Ausgabe des Objektes[7]. Zum Vergleich: Ein Dynamikzyklus benötigte bei diesem Modell eine Rechenzeit von 1,87 ms. Die graphische Ausgabe der Modelle vom KISMET-GEO-Typ 'ELADYN_NURBS' (Anhang B.2) kann wahlweise als NURBS, einfache Polyeder oder verfeinerte Polyeder erfolgen[8], der KISMET-GEO-Typ 'ELADYN_POLY' (Anhang B.3) ermöglicht eine kompaktere Definition, allerdings nur eine Ausgabe in einer Polyeder-Darstellung.

5.3 Schattierung, Lichtmodelle, Texturierung - Rendering[9]

Wesentlich zur Steigerung des Realismus in graphischen Simulationsszenarien ist die Nutzung von Rendering-Techniken aus der graphischen Datenverarbeitung. Diese bauen auf den geometrischen Modellen auf und stellen diese nach entsprechenden Vorgaben und Nebenbedingungen auf dem Bildschirm dar. Eine effiziente Nutzung muß die Möglichkeiten der Rechnerhardware sowie der Graphikbibliotheken berücksichtigen. Geeignet sind jedoch nur echtzeitfähige Verfahren, aufwendige Beleuchtungsmodelle wie Ray-Tracing (angesprochen in Kapitel 2) sind nicht praktikabel. An dieser Stelle sollen die Rendering-Techniken kurz angesprochen werden, die zum Verständnis der vorliegenden Arbeit wesentlich sind. Eine ausführliche Behandlung findet sich in [FDF90], [FvD84] sowie [SGI91]. Die Verfahren und Eigenschaften sind prinzipiell unabhängig vom physikalischen Modell. Sie können aber bei Interaktionen und insbesondere bei Modellmodifikationen beeinflußt werden, beispielsweise durch Anpassen von Farbverläufen (Facettenschwärzung, Texturanordnung).

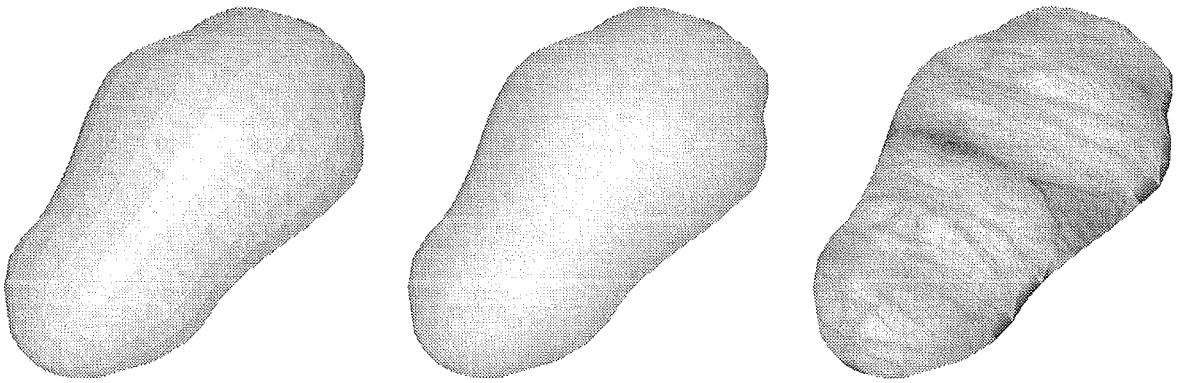
Für die Schattierungs- und Beleuchtungsmodelle ist die Festlegung verschiedener **Lichtquellen** mit spezifizierten Eigenschaften im Simulationsszenario wichtig. In der Anwendung als Endoskopie-Trainer ist - entsprechend den realen Gegebenheiten - die Definition einer gerichteten Punktlichtquelle vorne an der Endoskop-Optik sinnvoll[10].

[6] Auf einer Silicon Graphics Indigo Extreme, ein R4400-Prozessor, Modellgröße: 113 Knoten, 430 VE
[7] Bei NURBS der Maximalwert, da die Tesselationsgenauigkeit von der Darstellungsgröße (Blickwinkel) abhängt. Zugrundegelegt wird die GL-Tesselation (Darstellungsprimitiv 'nurbssurface').
[8] Strukturänderungen bei Modellmodifikationen sind in der NURBS-Darstellung nicht implementiert.
[9] Für 'Rendering' existiert keine passende deutsche Übersetzung (~ 'Darstellungsprozeß').
[10] Der Beobachterpunkt bei der perspekivischen Projektion des Szenarios ist ebenfalls an der Spitze der Endoskop-Optik lokalisiert.

Essentiell ist die Vorgabe der **Materialattribute,** welche die optischen Eigenschaften der Geometrie festlegen. Neben der Grundfarbe spielt hierbei insbesondere die Oberflächenbeschaffenheit eine große Rolle. Mit den Reflexionsparametern wird der reflektierte, absorbierte und transparente Anteil für diffuse oder direkte Lichteinwirkung festgelegt. Spiegelnde Reflexion täuscht beispielsweise eine feuchte Oberfläche vor.

Das **Schattierungsmodell** kann den visuellen Eindruck einer glatten und stetigen Oberfläche wiedergeben, insbesondere bei den größeren Facetten der Polyeder-Modelle. Bei der verwendeten Gouraud-Schattierung werden die Farbintensitäten in Abhängigkeit vom Beleuchtungsmodell an allen Polygoneckpunkten berechnet, die Farbintensitäten an den Polygonkanten und innerhalb der Facette werden linear interpoliert. Notwendig ist hierzu die Festlegung des Normalenvektors für jeden Polygoneckpunkt. Innerhalb des MESD-Verfahrens werden die Normalenvektoren für jeden G-Knoten berechnet (Kapitel 4) und in der Knotendomäne (Bild 5.5) abgelegt.

Der Einsatz von **Texturierung** kann den deformierbaren Körpern wie auch dem umgebenden Szenario ein realistisches Aussehen verleihen. Jedem Oberflächenpunkt wird ein Feldelement der Textur zugewiesen. Für jeden Polygoneckpunkt muß die Vorgabe der Texturkoordinaten explizit erfolgen, sie werden wie die Normalenvektoren in den Knotendomänen abgelegt. Die Zwischenwerte werden interpoliert. Bei NURBS vereinfacht sich die Definition, die dortigen Flächenparameter s, t dienen direkt als Argumente zur Bestimmung der Texturkoordinaten. Bild 5.9 zeigt das verfeinerte Polyedermodell unter Verwendung verschiedener Rendering-Techniken.



Ebene Schattierung | Gouraud-Schattierung | Gouraud & Textur

Bild 5.9: Darstellung eines Modells bei Gouraud-Schattierung und Texturierung

5.4 Zusammenfassung des Kapitels

Dieses Kapitel behandelt die graphische Repräsentation von Modellen auf Basis des MESD-Verfahrens, also die Visualisierung deformierbarer Körper. Es wurden hierbei zwei verschiedene Ansätze betrachtet und deren Vor- und Nachteile diskutiert: die Darstellung über Freiformflächen (NURBS) sowie die direkte polygonbasierte Darstellung. Eine Polyeder-Darstellung mit anschließender Polygonnetzverfeinerung durch ein schnelles Interpolationsverfahren stellt eine qualitativ ausreichende und gleichzeitig effiziente Lösung dar. Weiterhin wurden kurz verschiedene Rendering-Möglichkeiten angesprochen, die den Realismus eines graphischen Simulationsszenarios wesentlich erhöhen können.

6 Der Karlsruher Endoskopie-Trainer

In diesem Kapitel soll eine wichtige Anwendung der entwickelten Verfahren vorgestellt und erläutert werden. Der 'Karlsruher Endoskopie-Trainer' stellt eine echtzeitfähige Simulationsumgebung dar, die die Ausbildung in der Minimal-Invasiven Chirurgie sinnvoll unterstützen kann. Weitere mögliche Anwendungsfelder werden im nächsten Kapitel zusammengefaßt.

6.1 Minimal-Invasive Chirurgie, Systemanforderungen

In den letzten Jahren hat sich die Minimal-Invasive Chirurgie (MIC) fest in der chirurgischen Praxis etabliert [Bue95]. Hierbei wird der operative Eingriff mit Hilfe einer Endoskopkamera sowie langer, dünner Instrumente durchgeführt, die durch natürliche oder kleine künstliche Körperöffnungen in das Operationsgebiet eingeführt werden (Bild 6.1). Die hiermit verbundenen Vorteile für den Patienten, wie geringere Schmerzen und schnellere Genesung, bedingen jedoch gewichtige Nachteile für den Operateur: eingeschränktes Sichtfeld, beschränkte Beweglichkeit, fehlendes Tastgefühl sowie schwierige Handhabung der Instrumente.

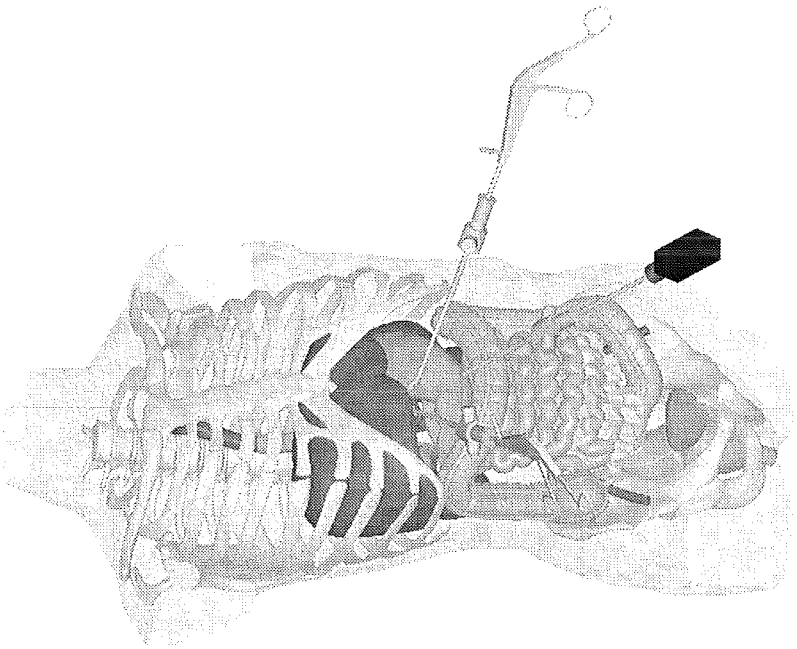


Bild 6.1: Simuliertes Szenario einer Laparoskopie

Die Technik der Minimal-Invasiven Chirurgie verlangt eine intensive Ausbildung der Operateure, um mit den gestiegenen Anforderungen vertraut zu werden. Mit Methoden der Computergraphik kann die Ausbildung und die Lösung weiterer Problemstellungen in der Minimal-Invasiven Chirurgie sinnvoll unterstützt und ergänzt werden [KN93, KN94, KKK95]. Ein erster Ansatz ist die möglichst realistische dreidimensionale Visualisierung der menschlichen Anatomie beziehungsweise des Operationsfeldes mit Organen, Gefäßen und Geweben [KKK94].

Für eine interaktive Simulationsumgebung ist es notwendig, in einem weiteren Schritt neben den geometrischen Formen auch das mechanische Verhalten von biologischem Gewebe nachzubilden. Weiterhin ist die Integration typischer chirurgischer Instrumente als funktionelles Rechnermodell erforderlich, mit denen der Bediener interaktiv Einfluß nehmen und die modellierten Gewebe manipulieren kann. Operationsvorgänge können so graphisch in Echtzeit simuliert werden ('Virtuelle Realität'). Der 'Karlsruher Endoskopie-Trainer' wurde unter diesen Vorgaben prototypisch für den Bereich der endoskopischen Bauch- und Unterleibschirurgie (Laparoskopie) entwickelt.

Im Vergleich zu den momentan durchgeführten Ausbildungsmethoden (Training an Tierpräparaten, Kunststoffnachbildungen oder auch am lebendigen Tier) kann dieser Ansatz die Ausbildung angehender MIC-Chirurgen wesentlich verbessern und effizient gestalten (Bild 6.2). Weiterhin ist auch bei schon praktizierenden Chirurgen eine stetige Weiterbildung und Übung unerläßlich, um eine Operation auch bei Eintreten von Komplikationen sicher und erfolgreich durchführen zu können. Eine Analogie kann hierbei durchaus zu Flugsimulatoren bei der Pilotenausbildung und -weiterbildung gesehen werden.

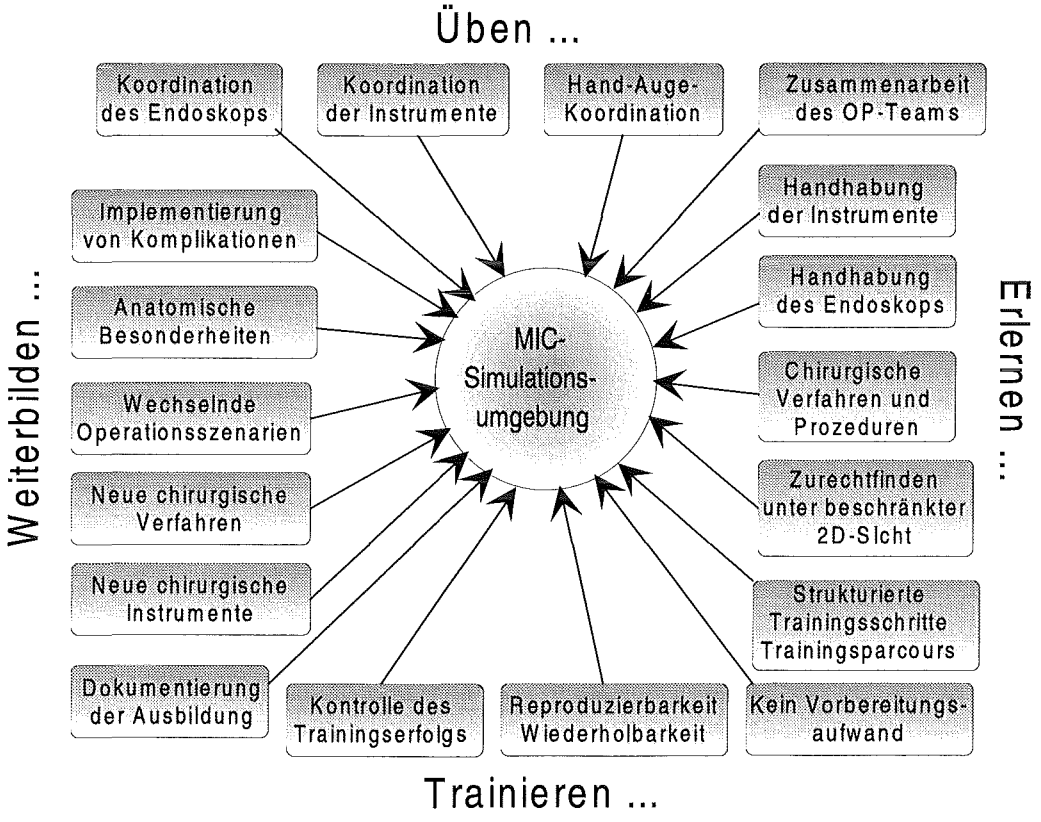


Bild 6.2: Möglichkeiten einer 'Virtual Reality'-Trainingsumgebung

Im medizinischen Bereich ergeben sich für den hier vorgestellten Ansatz einer rechnergestützten Simulationsumgebung weitere sinnvolle Einsatzgebiete und Applikationen, sowohl in der technischen Entwicklung als auch in der chirurgischen Praxis:

- Testumgebung für die Instrumenten- und Manipulatorenentwicklung
- Testumgebung für die MIC-Verfahrensentwicklung (Telemanipulation)
- Navigationsunterstützung während operativer Eingriffe (Zusatzinformationen durch 'synthetisches Sehen')
- Planung chirurgischer Eingriffe (mit patientenspezifischen Daten)
- Medizinische Ausbildung (Anatomiemodelle)

Verschiedene Arbeiten zur Entwicklung von Graphischen Simulationssystemen für medizinische Anwendungen sind in der Literatur bekannt, wie beispielsweise der vom Fraunhofer-Institut (IGD) entwickelte Arthroskopie-Simulator [ZMF95]. Dieser verwendet jedoch keine deformierbaren Objekte und wertet keine Interaktionen aus. Einen Ansatz für einen chirurgischen Simulator stellte Dumay [Dum95] vor, ein deformierbares Organmodell wurde von Cotin [CDC96] entwickelt. Simulatoren für die Laparoskopie werden in den USA von den Firmen Ixion [Hon96] und High Techsplanations [MRM96] entwickelt. Interaktive Modellmanipulationen und -modifikationen sind hier aber nur sehr beschränkt möglich.

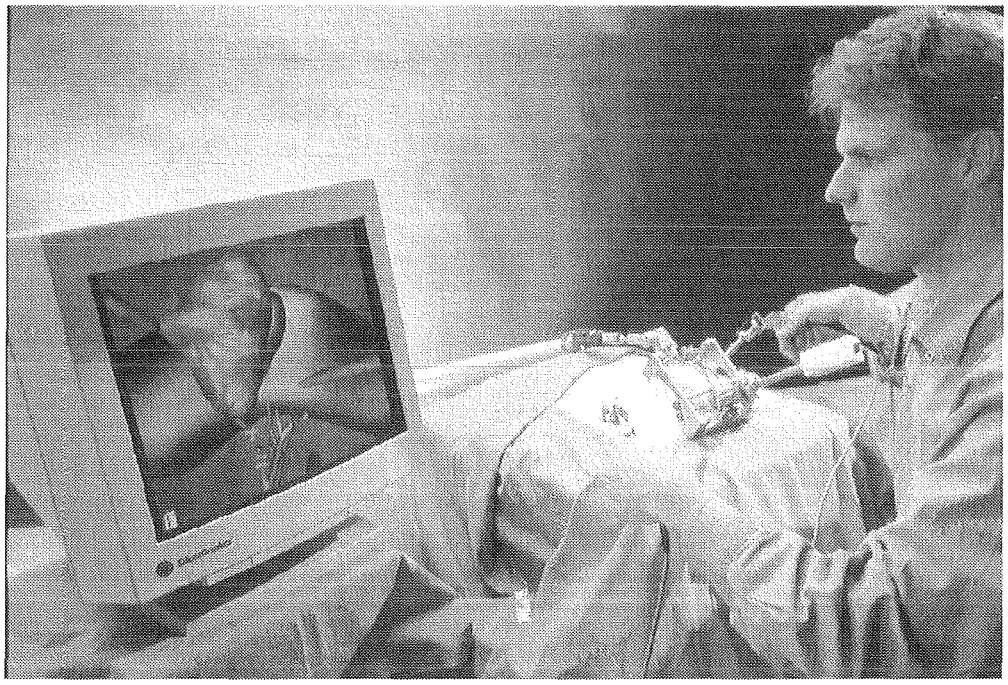


Bild 6.3: Der 'Karlsruher Endoskopie-Trainer'

6.2 Konzept des 'Karlsruher Endoskopie-Trainers'

Für echtzeitfähige graphische Simulationsumgebungen werden sehr hohe Anforderungen an die verwendete Hardware und die implementierte Software gestellt. Die in den vorangegangenen Kapiteln eingeführten Verfahren sind insbesondere unter dem Aspekt der größtmöglichen Effizienz hinsichtlich Rechenzeit und Datenverwaltung entwickelt worden. Deshalb sind sie für diese Applikation besonders geeignet.

Wie in Kapitel 2.4 schon erläutert, wurde für die Fernhantierungstechnik das Softwarepaket KISMET am Forschungszentrum Karlsruhe entwickelt [Kue91]. Aufgrund des gleichen Anforderungsprofils eignet sich KISMET prinzipiell auch für den Bereich der endoskopischen Chirurgie, insbesondere in Kombination mit den in dieser Arbeit entwickelten Verfahren.

Ein modellbasierter, hierarchischer Ansatz soll die Anpassung der virtuellen Szenarien an die verschiedenen medizinischen Einsatzfelder ermöglichen (Laparoskopie, Urologie, Gynäkologie, Neurochirurgie). Ziel ist eine möglichst realistische Simulation des Operationsfeldes, bei der die endoskopische Sicht vom Rechner nachgebildet wird. Der Echtzeitfähigkeit wird die entscheidende Bedeutung zugemessen, da der Benutzer über die modellierten Instrumente interaktiv in das Operationsfeld eingreift.

6.2.1 Aufbau und Struktur der Trainingsumgebung

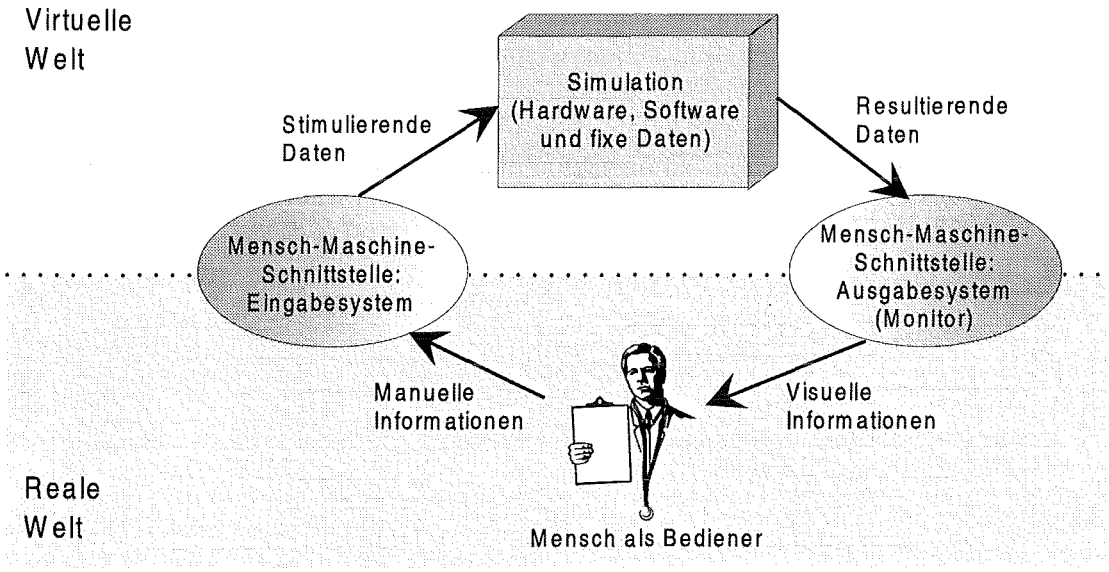


Bild 6.4: Trainingsumgebung als 'Man in the Loop'-Simulation

Die Simulationsumgebung wird durch eine 'Man in the Loop'-Simulation charakterisiert, also einen über den Menschen geschlossenen Regelkreis (Bild 6.4). Zwei Mensch-Maschine-Schnittstellen ermöglichen jeweils einen unidirektionalen Informationsfluß. Diese Ein/Ausgabesysteme versuchen, die originalen Gegebenheiten nachzubilden, um damit dem Bediener einen realitätsgetreuen Eindruck zu vermitteln. Das Trainingssystem besteht im wesentlichen aus folgenden Komponenten:

- Zentrale Einheit ist eine Hochleistungs-Graphikworkstation, auf der das modifizierte Simulationssystem KISMET betrieben wird (Bild 6.5). Das eigentliche Operationsfeld wird hierbei ausschließlich im Rechner generiert, die Berechnungen und Graphikausgabe erfolgen in Echtzeit. Für eine realistische Simulation wird eine Modelldatenbasis benötigt, die die geometrischen Formen und die physikalischen Eigenschaften der Gewebe, Organe und Gefäße sowie die Geometrie und Kinematik der Instrumente festlegt. Weiterhin definiert eine Wissensbasis das Verhalten bei Interaktionen, die Funktionalität der Instrumente sowie die Behandlung von Modellmanipulationen.

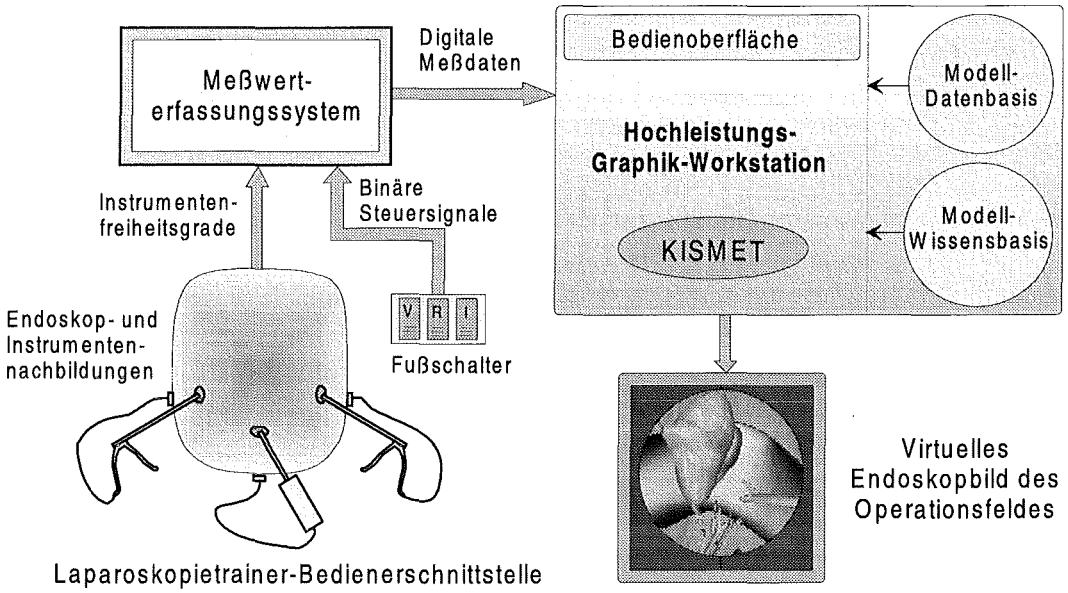


Bild 6.5: Struktur der Simulationsumgebung

☞ Als Benutzerschnittstelle dient ein Phantomgehäuse, das dem zu operierenden Körperteil äußerlich nachgebildet ist. Das Gehäuse dient zur Aufnahme der Endoskopnachbildung sowie der MIC-Instrumentenschäfte (Bild 6.6). Die Griffstücke werden von marktüblichen Instrumenten übernommen, um eine realistische Bedienung zu ermöglichen. Im Gehäuse werden die MIC-Instrumente in einem mechanischen Führungssystem geführt, das weiterhin die Erfassung der Auslenkung der Instrumente und Effektoren erlaubt (Bild 6.7). Die Freiheitsgrade werden über ein Sensorsystem einzeln aufgenommen, um auch kinematisch redundante Strukturen berücksichtigen zu können. Zusätzlich sind verschiedene Fußschalter vorhanden, über die sich chirurgische und allgemeine Funktionen aktivieren lassen. Die Umsetzung der Sensordaten und die Übertragung der Meßdaten zur Workstation wird durch ein PC-basiertes Meßwerterfassungssystem durchgeführt. Ein I/O-Prozeß übernimmt diese digitalen Daten und stellt sie KISMET in einem gemeinsamen Speicherbereich zur Verfügung.

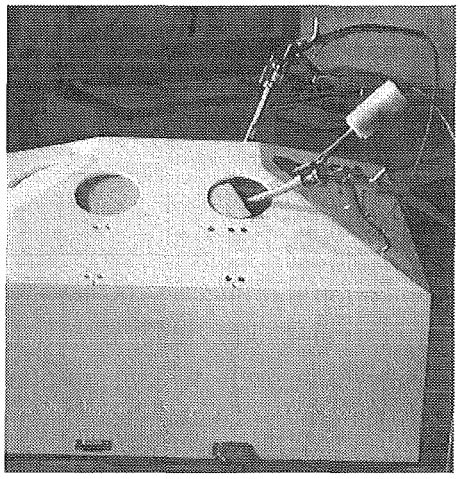


Bild 6.6: Bedienerschnittstelle

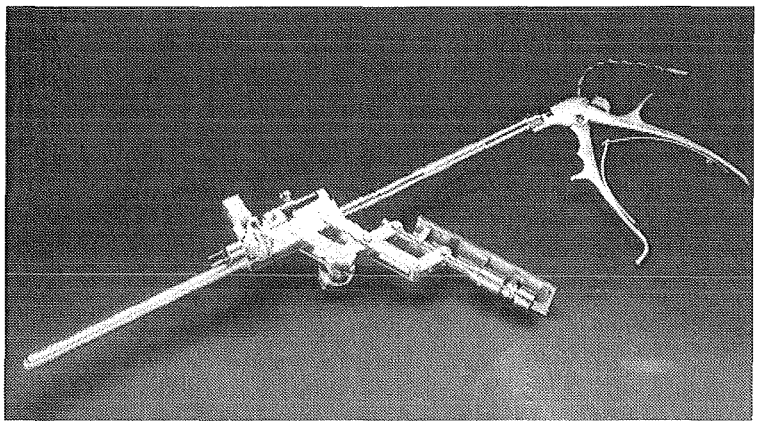


Bild 6.7: Instrument mit Führungssystem und Sensorik

Für eine direkte Interaktion und realistisches Empfinden beim Trainieren sind nur geringe Zeitverzögerungen von Stimulation (Bewegung der Instrumente durch Bediener) zur visuellen Reaktion (Wahrnehmung der Bewegung durch den Menschen) zulässig. Der Aufbau sollte deshalb für schnelle Datenverarbeitung und -übertragung optimiert werden. Die notwendigen

Prozesse sind entkoppelt und werden asynchron abgearbeitet, um eine möglichst hohe Bildwiederholrate und eine geringe Simulationstotzeit zu gewährleisten. Die Abschätzung des Zeitverhaltens erfolgt anhand einer Betrachtung der Signallauf- und Berechnungszeiten (Bild 6.8). Als Beispiel werden der Aufbau und die Modelle des Demonstrators (Bild 6.5, Kapitel 6.3) herangezogen.

Die Erfassung und A/D-Wandlung erfolgt mit einer Konvertierungsrate von maximal 30 kHz. Bei 16 analogen Kanälen[1] pro Karte beträgt die interne Erfassungszeit damit höchstens 2 ms. Die Datenübertragung erfolgt mit 38400 Bit/s über eine serielle RS232-Schnittstelle. Bei 32 Byte pro Datensatz[2] ergibt dies eine Transferrate von 150 Datensätzen pro Sekunde, mit Schnittstellenverwaltung also eine Übertragungszeit von etwa $\tau_{Ü}$=10 ms. Die gesamte Verzögerungszeit τ_V von der Meßwert-Anforderung bis zur Bereitstellung der Daten auf der Workstation beträgt ungefähr τ_V=19 ms (gemessener Wert). Hierin sind neben der Übertragungszeit $\tau_{Ü}$ auch die PC-Rechenzeiten zur Konvertierung und Kennlinienkorrektur (Linearisierung) enthalten.

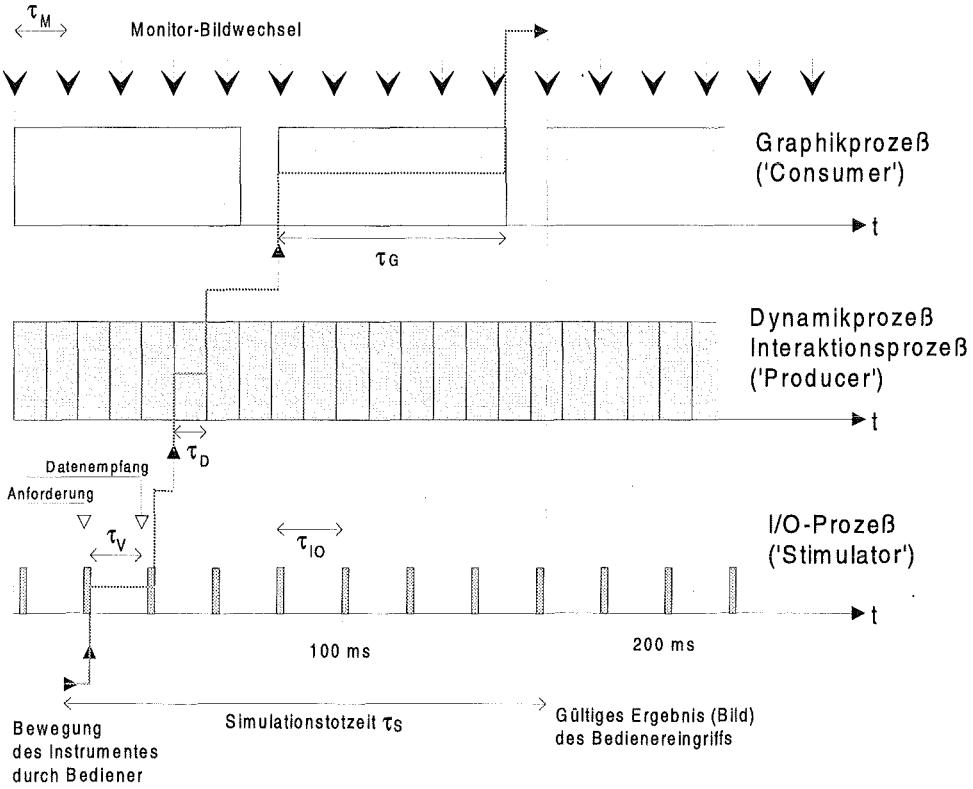


Bild 6.8: Zeitdiagramm der Simulationsumgebung

Die optimale Taktrate f_{IO} des I/O-Prozesses, der zur Kommunikation mit dem Meßwerterfassungssystem dient, wird durch die Verzögerungszeit τ_V bestimmt. Für eine minimale Erfassungstotzeit soll die Zykluszeit $\tau_{IO}=1/f_{IO}$ etwa der Verzögerungszeit τ_V entsprechen. Sinnvoll ist daher eine Taktrate von f_{IO}=50 Hz, die durchschnittliche Totzeit der Datenerfassung beträgt somit $\tau_T \approx \tau_{IO}$=20ms. Die Anforderung eines neuen Datensatzes erfolgt hierbei am Ende jedes I/O-Aufrufes, so daß bei Aktivierung zum nächsten Taktzyklus gerade wieder gültige Daten anliegen. Durch diese asynchrone Vorgehensweise belegt der I/O-Prozeß auch nur eine geringe Prozessorzeit.

[1] Für drei Instrumente mit je fünf Freiheitsgraden sind 15 Kanäle notwendig !
[2] Zwei Bytes pro Analog-Kanal (= 30) sowie je ein Byte für digitale Eingänge und Prüfsumme.

Der Monitorbildwechsel erfolgt mit f_M=60 Hz, die maximale Bildausgabeverzögerung beträgt also τ_M=16 ms. Für die Berechnung und Ausgabe der Graphik kann eine Zeit von τ_G=80 ms (Tabelle 6.1) angenommen werden, so daß die Bildwiederholrate[3] bei etwa f_S=12 Hz liegt. Ein Dynamikzyklus wird innerhalb von τ_D=7 ms berechnet. Die Simulationstotzeit als Summe der einzelnen Verarbeitungszeiten liegt somit durchschnittlich bei etwa τ_S=0,14 s. Dieser Wert ist für eine realistische Interaktion ausreichend. Durch einen modularen Aufbau ist die Erweiterbarkeit der Bedienschnittstelle und des Auswertesystems gewährleistet, so daß weitere Instrumente und Freiheitsgrade berücksichtigt werden können.

6.2.2 Systemintegration und Softwaremodule

Die in dieser Arbeit entwickelten Verfahren wurden in das Simulationssystem KISMET integriert und mit den schon vorhandenen Modulen gekoppelt. Ein Blockdiagramm mit der funktionalen Struktur der Softwaremodule und der Darstellung des Datenflusses zeigt Bild 6.9. Wichtig bei der Integration in KISMET war eine genaue Definition der Schnittstellen und der Ein-/Ausgabedaten, um den universellen modularen Charakter des Systems sicherzustellen.

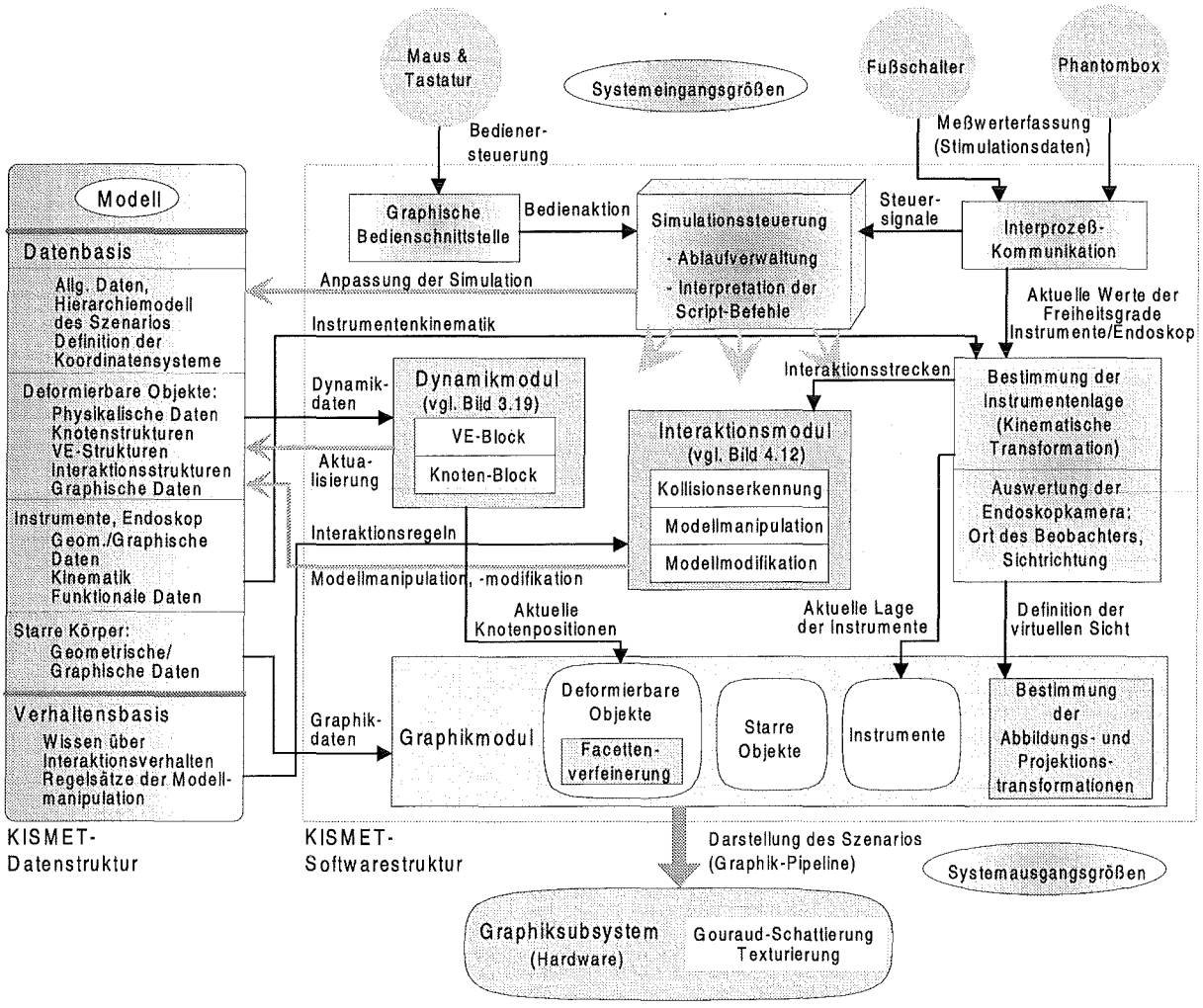


Bild 6.9: Softwarestruktur der Simulationsumgebung

[3] Der Simulationsbildwechsel und damit die resultierende Bildwiederholrate erfolgt aufgrund interner Graphik-Synchronisation immer mit einem ganzzahligen Teil der Monitor-Bildwechselfrequenz: $f_S = f_M/n$.

Für die Gewährleistung der Echtzeitfähigkeit ist der asynchrone Ablauf einzelner Module innerhalb des Gesamtsystems vorteilhaft. Eine sinnvolle Aufteilung der Module auf die zur Verfügung stehenden Prozessoren unterstützt dieses Konzept. Die Aspekte der Parallelisierung werden ausführlich in Kapitel 3.4.3 behandelt. Implementiert wurden die einzelnen Module als geschwindigkeitsoptimierter Code in der Programmiersprache C.

Die Datenspeicherung zur Simulationslaufzeit erfolgt in zusammenhängenden, verzeigerten Listen, um eine optimale Ausnutzung des Cache-Speichers zu gewährleisten und damit die Prozessorauslastung zu optimieren (Bild 3.28, Bild 5.5). Die entsprechenden Datenstrukturen werden bei der Initialisierung der Simulation vollständig generiert und zur Laufzeit strukturell nicht mehr modifiziert.

Für eine Leistungsanalyse werden zur Messung der Rechenzeiten der einzelnen Softwaremodule zwei Beispielmodelle herangezogen[4].

Modell 1: Demonstrationsmodell 'Gallenblase & Gallengang', siehe Bild 6.11:
Objektprimitiv 'Organ' mit Netzmatrix 18x12, 195 Knoten,
756 Verbindungselementen, 1536 Facetten sowie Objektprimitiv 'Rohr' mit Netzmatrix 10x12, 130 Knoten, 561 Verbindungselementen, 864 Facetten;
verknüpft mit 12 Hyperknoten

Modell 2 : Gleiche geometrische Form wie Modell 1, aber mit erhöhter Modellgenauigkeit (etwa doppelte Anzahl von Netzkomponenten):
Objektprimitiv 'Organ' mit Netzmatrix 22x20, 443 Knoten,
1740 Verbindungselementen, 3200 Facetten sowie Objektprimitiv 'Rohr' mit Netzmatrix 13x20, 273 Knoten, 992 Verbindungselementen, 1920 Facetten;
verknüpft mit 20 Hyperknoten.

Tabelle 6.1: Rechenzeiten der Software-Module

Softwaremodul	Modell 1 (ms)	Modell 2 (ms)
Dynamikmodul, VE-Block (vgl. Bild 3.19): Auswertung der Verbindungselemente, Berechnung der Knotenkraftvektoren	3,31	6,75
Dynamikmodul, Knoten-Block (vgl. Bild 3.19): Numerische Integration, Bestimmung der Knotenzustandsvektoren	1,66	3,48
Kinematische Transformationen aus den Instrumenten-Meßwerten	0,85	0,87
Interaktionsmodul (vgl. Bild 4.12) : Kollisionserkennung, Interaktionsbehandlung		
- mimimal (keine Interaktion)	0,281	0,289
- maximal (vielfache Interaktion)	1,277	1,593
Zykluszeit Prozessor 1	6,1 - 7,1	11,7 - 13,3
Simulationssteuerung, allgemeine Transformationen, Abfrage der graphischen Bedienerschnittstelle	27,6	28,1
Verfeinerung und graphische Ausgabe der deformierbaren Objekte	10,8	22,3
Graphische Ausgabe sonstiger Objekte im Szenario, Instrumente	50,5	50,4
Zykluszeit Prozessor 2	87,3	101,2

Die Integrationsschrittweite der numerischen Lösungsverfahren ergibt sich direkt aus der Zykluszeit des ersten Prozessors (h ≈ 6-13 ms), während die Bildwiederholrate aus der

[4] Als Referenzrechner dient eine Silicon Graphics 'Onyx' mit VTX-Graphiksubsystem und zwei MIPS R4400-Prozessoren (200 MHz). Die Zeiten sind Mittelwerte aus zehn Messungen mit geringer Varianz.

Zykluszeit des zweiten Prozessors resultiert ($f_S \approx$ 9-12 Hz). Die Rechenzeiten einiger Softwaremodule sind unabhängig von der Größe der MESD-Modelle (Simulationssteuerung, kinematische Transformation, graphische Ausgabe starrer Objekte). Die Module zur Berechnung und Darstellung der deformierbaren Objekte besitzen einen linearen Zusammenhang zwischen Rechenzeit und Modellgröße, wie es auch das MESD-Verfahren erwarten läßt. Durch die Hierarchisierung der Kollisionserkennung sind die Bearbeitungszeiten des Interaktionsmoduls stark von der Art der momentanen Interaktion abhängig. Aus dem gleichen Grund wirkt sich eine Erhöhung der Modellgröße nicht wesentlich auf die notwendige Zeit zur Detektion einer Kollision aus.

6.2.3 Modellbildungskonzept

Bei der Realisierung der Simulationsumgebung ist von besonderer Wichtigkeit, menschliches Gewebe realistisch nachzubilden. Dies führt zu einem System von 'deformierbaren Objekten' mit originalgetreuen geometrischen Formen und möglichst natürlichem Verhalten. Ein weiterer wichtiger Punkt ist die realistische Simulation der Interaktion zwischen den deformierbaren Objekten und den Instrumenten sowie die Manipulation der virtuellen Gewebe in Echtzeit. Diese Anforderungen verlangen ein neues, homogenes Modellbildungskonzept, um eine effiziente Simulation zu erreichen. Für den Kern der Simulationsumgebung müssen die in dieser Arbeit behandelten Teilaspekte verschmolzen werden, wie auch Bild 6.10 verdeutlicht.

Datenbeschaffung

Wichtig für den Trainingserfolg und damit die Akzeptanz einer rechnerbasierten Trainingsumgebung ist eine gute Nachbildung des realen Operationsszenarios. Deshalb ist der Einsatz von realen Referenzdaten notwendig. Während dies bei technischen Systemen qualitativ und quantitativ möglich ist, ergibt sich bei biologischen Systemen wie der menschlichen Anatomie das Problem der Datenbeschaffung. Jeder Mensch ist verschieden, seine anatomischen Eigenschaften sind stets unterscheidbar und deshalb nicht allgemein bestimmbar [DK95]. Es muß daher ein 'Standardmensch' herangezogen werden, dessen Eigenschaften als Referenzmodell der Trainingsumgebung dienen[5].

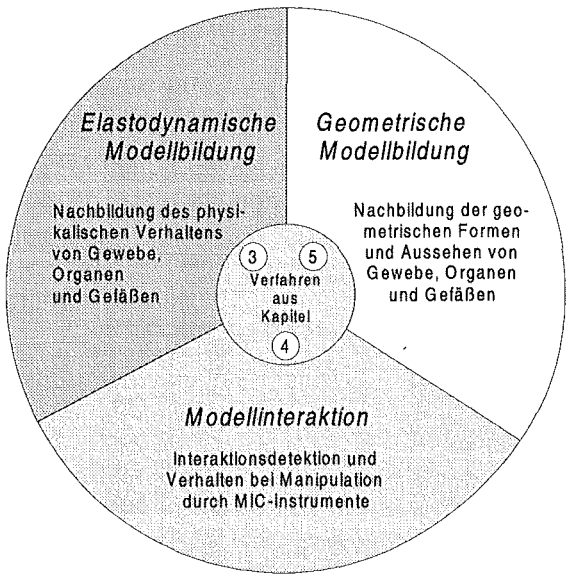


Bild 6.10: Modellbildungsschema der Simulationsumgebung

[5] Solange keine weitgehend automatisierte Modellerzeugung möglich ist.

Besonders bei der Beschreibung der mechanischen Eigenschaften von organischem Material ergeben sich große Schwierigkeiten. Wissenschaftliche Untersuchungen zu dieser Problematik wurden vielfach durchgeführt [KGE75, Lar86], konkrete Daten beschränken sich jedoch im wesentlichen auf Haut [Lar86], Skelett und Muskelgewebe [NH94]. Eine quantitative Bestimmung von Parametern ist sehr schwierig, da nichtdurchblutete Organe ein anderes physikalisches Verhalten zeigen als lebendige, im Körper befindliche Gewebe. Außerdem variieren die mechanischen Eigenschaften stark, beispielsweise in Abhängigkeit vom Alter, Körperbau oder Vorgeschichte des Organs (Entzündungen). Deshalb kommt es im Rahmen dieser Anwendung nicht auf eine exakte, quantitative Simulation der physikalischen Eigenschaften an, sondern auf eine qualitativ korrekte Repräsentation des Operationsfeldes.

Die Modellbildung läßt sich durch Methoden fortentwickeln, mit denen aus bildgebenden Verfahren der Medizin (Tomographie) neben den geometrischen Daten auch physikalische Eigenschaften und das Verhalten von Gewebe abgeleitet werden kann. Mit solchen patientenspezifischen Daten läßt sich die vorgestellte Simulationsumgebung auch zur speziellen Operationsvorbereitung einsetzen, also zum Einstudieren einer Operation vor dem Ernstfall.

Tabelle 6.2: Datenaquisition für die Modellbildung - **Instrumente**

Geometrie	Umsetzung von CAD-Daten der Originalinstrumente
Kinematik	Vereinfachte Ableitung der Kinematik von Originalinstrumenten (Innere Kinematik der Effektorbetätigung unrelevant)
Funktionalität	Je nach Typ des Instrumentes haben Objektmanipulationen verschiedene Auswirkungen, zumindest bei Betätigung des Effektors. In einer Regelbasis wird spezifisches 'Metawissen' abgelegt, aufgrund dessen die Aktivierung der verschiedenen, instrumentenabhängigen Manipulationsmodule erfolgt.

Tabelle 6.3: Datenaquisition für die Modellbildung - **Organisches Gewebe**

Geometrie	CAD-modelliert (Freiformflächen, 'Softimage'), abgeleitet aus Anatomieatlanten und Anatomiestudien, auch mit Hilfe von visuellen Tomographiedatensätzen (nicht automatisiert)
Aussehen	Einsatz von Texturen aus realen Endoskopbildern, nachbearbeitet; Reflektionseigenschaften qualitativ angenähert
Mechanische Daten	nur schwer meßbar/erfaßbar, deshalb qualitative Beschreibung aus Erfahrungswerten, empirischer Abgleich von 'Experten' (iterativer Prozeß)
Manipulationsverhalten	Ablegen von vordefinierten Eigenschaften des spezifischen Manipulationsverhaltens in einem regelbasierten Verhaltensmodell

Aus Rechenzeitgründen werden nicht alle Objekte eines Operationsszenarios elastodynamisch modelliert, sondern nur diejenigen Organe und Gewebe, die im Rahmen der operativen Tätigkeiten manipuliert werden. Andere anatomische Strukturen werden rein geometrisch dargestellt. Sie dienen zur Erhöhung der visuellen Realitätstreue des Szenarios und lassen sich deshalb vom Bediener nicht interaktiv manipulieren.

6.3 Praktischer Einsatz - Demonstrator

Ein prototypisches Szenario einer Cholezystektomie (Gallenblasenentfernung) wurde in einem Demonstrator realisiert. Hierin sind sowohl die Gallenblase als auch der Ductus Cysticus (Teil des Gallenganges) in ihren geometrischen Formen und ihrem physikalischen Verhalten nachgebildet [ZDS92]. Auf einer Hochleistungs-Grafikworkstation Silicon Graphics 'Onyx' ist mit diesem Modell eine Bildwiederholrate von bis zu 12 Bildern/sec erreichbar (Bild 6.11).

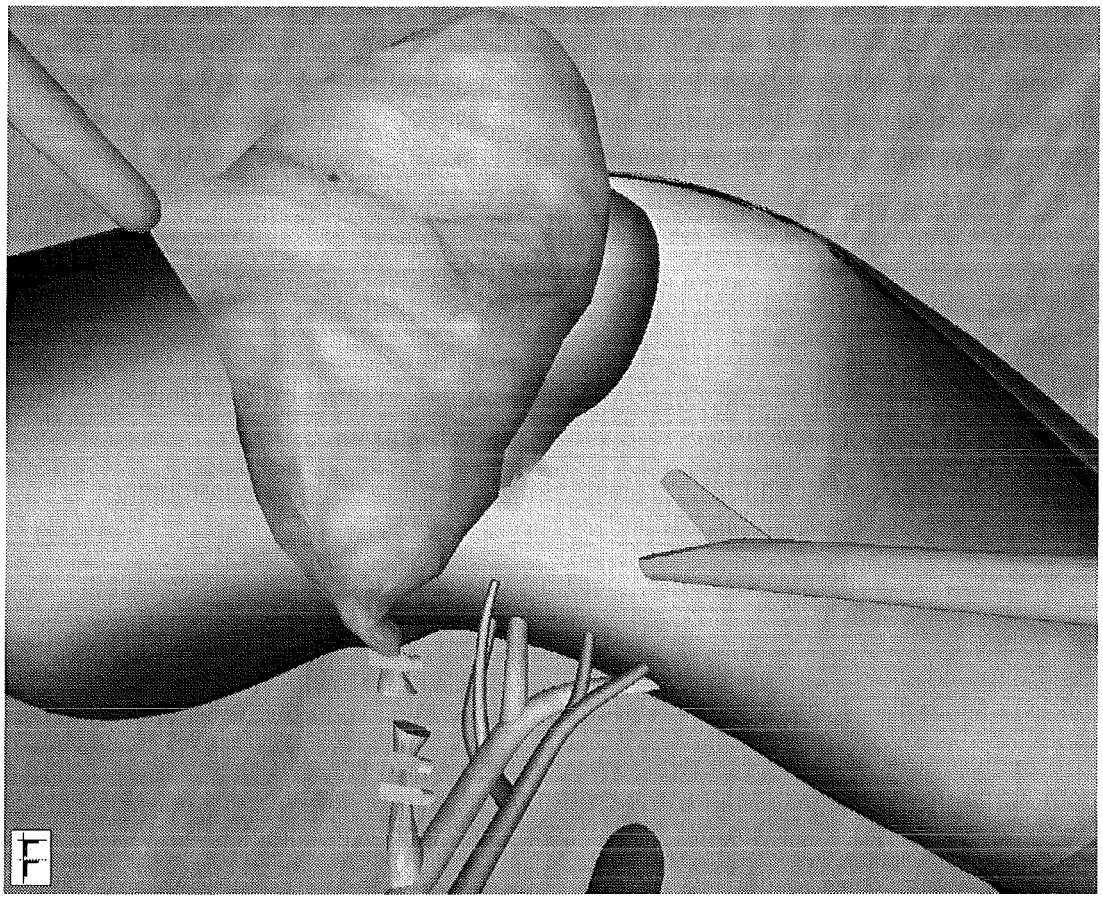


Bild 6.11: Simulierte Endoskopsicht einer Cholezystektomie

Durch aufprojezierte Texturen (erzeugt aus anatomischen Fotografien [GPS93]) sowie die zusätzliche Darstellung weiterer Organstrukturen (Leber, Gefäße) wird der optische Realismus erhöht. Eine weitere wichtige Option ist die stereographische Darstellung des virtuellen Endoskopbildes mit Hilfe einer Shutterbrille, um den Eindruck eines dreidimensionalen Bildes zu gewinnen.

Mit Hilfe diverser MIC-Instrumente (Greifer, Clip-Applikator, Schere, Koagulationshaken) ist eine interaktive Manipulation der modellierten Organe möglich. Hierzu erfolgt eine direkte Lagevorgabe des Instrumentes und des Endoskopes durch den Phantomtrainer. Der simulierte Clip-Applikator ermöglicht das Setzen von virtuellen Klemmen am Gallengang; mit der modellierten Schere läßt sich virtuelles Gewebe einschneiden und auch durchtrennen. Der Koagulationshaken wird zum Präparieren von Gewebestrukturen verwendet. Mit einem Fußschalter läßt sich das simulierte Auftrennen von Gewebe mittels Hochfrequenzstrom aktivieren.

Die prinzipielle Eignung des 'Karlsruher Endoskopie-Trainers' zur Unterstützung der Ausbildung und des Trainings von MIC-Chirurgen konnte anhand des Demonstrators gezeigt werden [KKN95]. Bei diversen Vorführungen und Präsentationen wurde das System vorgestellt, die Resonanz bei Chirurgen als späteren Anwendern war positiv. Im Oktober 1996 wurde das erste Trainingssystem für den praktischen Einsatz im 'Trainingszentrum für Minimal-Invasive Chirurgie' an der Universitätsklinik Tübingen installiert.

Die Genauigkeit der Simulation, die Korrektheit der Interaktionen sowie die Anzahl der elastodynamischen Objekte im Operationsfeld und damit die resultierende Realitätsnähe hängt in erster Linie von der zur Verfügung stehenden Rechnerleistung ab. Mit dem stetigen Fortschritt in der Computertechnologie und gleichzeitig günstigerem Preis/Leistungs-Verhältnis wird eine Verbesserung der Realitätstreue und ein weiterer Ausbau der simulatorischen Möglichkeiten der Trainingsumgebung einhergehen.

Als sinnvolle Erweiterungen und Ergänzungen des Systems in nächster Zukunft werden folgende Punkte angesehen:

- Kraftreflektierende Eingabegeräte: Rückkopplung der resultierenden Instrumentenkraft, Gewinnung taktiler Informationen während eines Operationsvorgangs [BGT96]
- Implementierung von biologischen Folgeereignissen: Simulation von Bluten, Gallenflüssigkeit, Pulsieren von Gefäßen (Aktivelemente)
- Implementierung weiterer chirurgischer Aktionen: Nähen mit Simulation des Fadenverhaltens, Setzen eines Katheders
- Anpassung der Bedienoberfläche an chirurgische Erfordernisse, Funktionen zur Trainingskontrolle
- Teilautomatische Segmentation von Patientendaten mit Erstellung von entsprechenden MESD-Modellen, Erweiterung der Modellierwerkzeuge (Parameterabschätzung)

Nach Ansicht des Autors werden rechnerbasierte Simulationssysteme in Zukunft eine große Rolle in der chirurgischen Aus- und Weiterbildung spielen und diese entscheidend verbessern. Erste Studien aus den USA [MGJ96, JBW96] belegen diese Aussage.

7 Zusammenfassung und Ausblick

Nach der zusammenfassenden Diskussion der vorliegenden Ergebnisse sollen in einem Ausblick einige weitere mögliche Anwendungsfelder der entwickelten Methoden angesprochen werden. Verschiedene Applikationen, welche die Simulationsumgebung für die Minimal-Invasive Chirurgie betreffen, sind schon im vorigen Kapitel aufgeführt.

7.1 Diskussion der Ergebnisse

In den vergangenen Kapiteln wurde das neuartige Konzept und bisher unveröffentlichte Methoden für die Modellbildung und die Echtzeitsimulation deformierbarer Körper vorgestellt. Weiterhin wurden Verfahren der Interaktion und Modellmodifikation präsentiert. Alle Methoden wurden im Rahmen der vorliegenden Arbeit auf einer Graphik-Workstation implementiert, getestet und verifiziert. Als prototypische Applikation diente hierbei der in Kapitel 6 ausführlich erläuterte 'Karlsruher Endoskopie-Trainer'.

- Dynamische Systeme auf Basis von nodalen Netzen können als grobe Vereinfachung der klassischen Finite-Elemente-Methode angesehen werden. Das MESD-Verfahren wiederum ist eine spezielle Ausprägung dieser Systeme, das eine besonders umfassende und effiziente Simulation mechanischer Sachverhalte ermöglicht. Dies beinhaltet eine klare, hierarchische Struktur der Modellbildung deformierbarer Objekte, sehr schnelle und damit echtzeitfähige numerische Behandlung sowie die Möglichkeit der Modifikation der Modelle zur Simulationslaufzeit.
- Das Verfahren ermöglicht implizit die Kopplung verschiedener dynamischer Objekte, die Definition starrer Körper mit dynamischen Attributen und damit eine globale dynamische Verknüpfung der Komponenten des Simulationsszenarios.
- Durch eine effiziente Kollisionserkennung wird ein interaktives Eingreifen des Bedieners in das Simulationsszenario ermöglicht. Durch entsprechende Manipulation wird eine Modellmodifikation ausgelöst, die je nach Art der Interaktion bestimmten Regeln und Verhaltensvorschriften folgt. Die Auswertung ist auf das Anwendungsgebiet der Minimal-Invasiven Chirurgie abgestimmt, die in diesem Bereich üblichen Instrumente werden für die Kollisionsdetektion abstrahiert. Implementiert wurden die Simulation von chirugischen Tätigkeiten wie das Greifen, Schneiden, Koagulieren und Setzen von Klemmen.
- Durch die Trennung von geometrischer und physikalischer Modellbildung, sowohl im Konzept als auch in der Datenstruktur, sind für die graphische Repräsentation deformierbarer Objekte prinzipiell verschiedene Ansätze möglich. Freiformflächen (NURBS) zeichnen sich hierbei durch eine stets glatte Darstellung und gute visuelle

Qualität aus. Der hohe Aufwand bei Berechnung und graphischer Ausgabe sowie die sehr komplexen Verfahren bei Modellmodifikationen lassen momentan jedoch nur einen beschränkten Einsatz zu. Eine direkte Graphikausgabe als polygonale Flächen vermindert zwar die optische Güte, ist jedoch wesentlich schneller und universeller modifizierbar. Durch Verfeinerung des polygonalen Netzes mittels Interpolation von Zwischenpunkten und Einsatz von Gouraud-Schattierung wird eine ausreichende Darstellungqualität erzielt.

Der vorgestellte modellbasierte Ansatz gestattet die Übertragung der erläuterten Methoden bei der Realisierung von Trainingsszenarien und zahlreichen weiteren chirurgischen Aufgabenstellungen. Der Faktor, der in entscheidendem Maße die Qualität der Simulation beeinflußt, ist sicherlich die Rechen- und Graphikausgabezeit und damit die zur Verfügung stehende Rechnerhardware. Die Eigenschaften der deformierbaren Objekte und das Interaktionsverhalten wird bei kleineren Simulationsschrittweiten und feinerem nodalen Netz genauer, stabiler und damit realistischer. In das Modellszenario können gleichzeitig mehr Details integriert werden, die Bildwiederholrate kann gesteigert werden und so den visuellen Eindruck und die Immersion verbessern. Werden die Erfahrungen der letzten Jahre zugrunde gelegt, so kann in den nächsten Jahren mit einer wesentlichen Steigerung der Leistungsfähigkeit von Rechnersystemen und damit mit einer Erhöhung der Qualität der Simulation gerechnet werden.

7.2 Weitere mögliche Anwendungen

Nodale Netzmodelle werden in der Animationstechnik schon geraume Zeit eingesetzt. Im Kapitel 'Stand der Technik' wurde darauf schon ausführlich eingegangen. Die in dieser Arbeit entwickelten Verfahren zeichnen sich insbesondere durch Echtzeitfähigkeit und interaktive Manipulierbarkeit aus. Deshalb sollen mögliche Anwendungsfelder speziell für diese Anforderungen vorgestellt werden. Absichtlich wird hierbei auch ein Blick in die Zukunft gewagt - ohne Anspruch auf Vollständigkeit.

Interaktive Computergraphik - 'Virtual Reality'-Anwendungen

'Virtual Reality'- das 'Eintauchen' (Immersion) in eine künstlich erzeugte Welt hat inzwischen Eingang in viele Anwendungsbereiche gefunden. Die angestrebte Realitätsnähe ist jedoch vielfach nur beschränkt gegeben. Während der eigentliche visuelle Eindruck inzwischen hauptsächlich von der Modellgröße und Modellgüte sowie der zur Verfügung stehenden Rechnerleistung abhängt, sind bei der physikalischen Repräsentation der Objekte, den Interaktionsverfahren sowie der taktilen Stimulation viele prinzipielle Verbesserungen möglich. Dies betrifft auch die Technik der angewendeten Ein- und Ausgabesysteme.

Die in dieser Arbeit behandelten Methoden können bei verschiedensten VR-Anwendungen den Realismus verbessern und damit den Grad der Immersion erhöhen. Natürliche Verformungen und Objektmodifikationen tragen hierzu wesentlich bei. Das MESD-Verfahren kann hierfür als Grundlage der Modellbildung und Simulation dienen. Somit können Szenarien, die bisher nur durch Animationen darstellbar waren, auch auf Echtzeit-Simulationen übertragen werden. Ein erster Ansatz ist die Simulation von Kleidern und Tüchern in virtuellen Welten [VT95].

Echtzeitfähige Interaktionstechniken sind für ein natürliches Verhalten in einer virtuellen Umgebung besonders wichtig. Wenn der Mensch in die synthetische Welt eingreift, müssen die resultierenden Auswirkungen realistisch simuliert und dargestellt werden. Die ausgeübten Kräfte auf ein Objekt werden berechnet und können als Basis für eine entsprechende Kraftrückkopplung verwendet werden.

Simulation deformierbarer Objekte in komplexen technischen Systemen

In Technik und Wissenschaft ist es mittlerweile Standard, komplexe Systeme durch Simulationen zu erfassen und damit die Wirkungsweise und Zusammenhänge zu untersuchen und verständlich zu machen. Bei der Entwicklung technischer Systeme kann die vorhergehende Simulation helfen, Fehler und Probleme schon in der Planungs- und Konstruktionsphase zu entdecken und vor dem Prototypbau zu beheben. Dieses Vorgehen verringert sowohl die Entwicklungszeit eines Produktes als auch die Entwicklungskosten. VR-Technologien verbessern den Entwicklungszyklus und dienen als Grundlage für das 'Rapid Prototyping' [Sch96].

Zur Optimierung eines technischen Systems können Konstruktion, Abmessungen und Parameter in der Simulation variiert werden, um die spezifischen Anforderungen zu erfüllen und ideale Eigenschaften zu finden. Eine qualitative und quantitative Aussage im systemtheoretischen Sinne erfolgt über die Beziehung von Eingangs- und Ausgangsgrößen, wobei das System mit charakteristischen Eingangsgrößen angeregt wird. Will man die Optimierungsparameter interaktiv zur Simulationslaufzeit verändern, so ist eine Simulation in Echtzeit notwendig.

Die vorgestellten Methoden eignen sich sehr gut für diesen Einsatz, da die Objektparameter jederzeit interaktiv geändert werden können. Eine solche Anwendung ist beispielsweise im Automobilbau denkbar: im Fahrzeug sind viele flexible Leitungen und Schläuche verlegt, insbesondere im Motorraum. Da aufgrund der Motorvibrationen und Schwingungen durch die Fahrbahnbeschaffenheit eine stetige mechanische Belastung auf die Leitungen einwirkt, sind diese besonders dem Verschleiß ausgesetzt. Die Simulation kann hier die Lösung folgender Fragestellungen unterstützen: Wie optimiert man die Verlegung ? Wie und wo befestige ich die Leitungen ? Welche Materialien verwende ich für Schläuche und für Befestigungen ? Gibt es bei bestimmten Anregungen und Belastungen Reibungsstellen mit starren Teilen ?

Neben dem Einsatz zur Unterstützung der Produktentwicklung kann das MESD-Verfahren auch Finite-Elemente-Methoden ersetzen und damit entsprechende mechanische Sachverhalte in Echtzeit simulieren und visualisieren - natürlich unter Verlust an Genauigkeit und mathematischer Exaktheit. Deshalb ist diese Methodik sicherlich nur für diejenigen Einsatzfälle praktikabel, bei denen es weniger auf Genauigkeit als auf Geschwindigkeit ankommt.

Nachbearbeitung von rekonstruierten Objekten

Verschiedene bildgebende Verfahren der Medizin verwenden physikalische Methoden, um strukturelle Daten aus dreidimensionalen Objekten zu gewinnen [HS85]. Beispiele hierfür sind die Magnetresonanz- oder Röntgentomographie (MRI, CT) sowie Ultraschall-Systeme. Bedingt durch die Auflösung der Verfahren, das Rauschen der physikalischen Meßwerte und prinzipielle Erfassungsfehler sind die resultierenden Datensätze verfälscht, beispielsweise können sie ein unstetiges Verhalten zeigen (Artefakte).

Tomographiedaten liegen meist in Form äquidistanter und paralleler Schnittbilder vor [HK96]. Sollen Objekte in ihrer geometrischen Form rekonstruiert werden, so werden Segmentierungsverfahren eingesetzt, die aus dem Datensatz bestimmte Merkmale extrahieren und Gebiete gleicher oder ähnlicher Merkmale zusammenfassen [Hoe87]. Aus diesen Klassen kann dann die Kontur der Objekte und damit ihre Oberfläche gewonnen werden [Kli94]. Bild 7.1 zeigt eine Schnittebene aus einem MRI-Datensatz, bei dem das Großhirn extrahiert und als helle Struktur dargestellt wird.

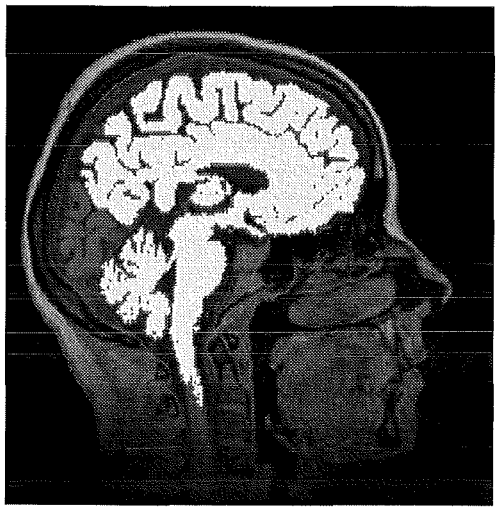


Bild 7.1: Teilsegmentierter MR-Tomographie-Datensatz

Probleme bereiten jedoch die oben angesprochenen Datenfehler, die zu fehlerhaften und sehr unplausiblen Rekonstruktionen führen können. Durch entsprechende Filterung der Geometriedaten lassen sich diese Ungenauigkeiten reduzieren [GMM96, NPW96]. Ein Ansatz hierzu ist die Glättung der Daten durch Aufprojektion eines nodalen Netzes auf die Objektkonturen. Das zu rekonstruierende Objekt wird hierbei als deformierbares System aufgefaßt, das einen Zustand minimaler Oberflächenenergie anstrebt. Merkmalspunkte des Objektes werden hierbei durch ein elastisches Netz verknüpft [TK95, RB96].

Das vorgestellte MESD-Verfahren kann aufgrund seiner Echtzeitfähigkeit die Filterung der Rohdaten wesentlich beschleunigen und effizienter gestalten. So kann bei einer schnellen, automatischen Aktualisierung der Bilddaten eine Objektglättung vor jeder Darstellung durchgeführt werden. Ein Applikationsbeispiel wäre die 'On-line'-Navigationsunterstützung während des Operationsablaufs, mit Hilfe derer zusätzliche, wichtige Informationen durch 'synthetisches Sehen' geliefert werden kann. Auch bei Mustererkennungsverfahren in der Bildverarbeitung könnten hiermit Verbesserungen erzielt werden, beispielsweise bei visuellen Prüfsystemen [PS90].

Es liegt nahe, die so gewonnenen Netze direkt als Grundlage für die elastodynamische Modellierung der in Kapitel 6 beschriebenen Trainingsumgebung heranzuziehen [Wat92]. Hierzu müßte jedoch eine Anpassung des Netzes durchgeführt werden, um die Echtzeitbedingung zu erfüllen. Außerdem ist es notwendig, daß auch noch weitere Informationen über die mechanischen Eigenschaften und das physikalische Verhalten in das Modell einfließen. Arbeiten hierzu sind am Forschungszentrum Karlsruhe aufgenommen worden.

7.3 Ausblick und Entwicklungspotential

Nichts ist vollkommen - dies gilt natürlich auch für die Thematik dieser Arbeit, wie der Autor auch bei manch einem Problem erfahren mußte. Bei vielen Teilfragen sind neue Ideen aufgekeimt, deren Behandlung aber über den Rahmen und die Ziele der Arbeit hinausgegangen wären. Deswegen ist noch viel wissenschaftliches Potential für tangierende und aufbauende Forschungen vorhanden. Am Anfang dieses Kapitels wurden schon einige Anwendungsrichtungen aufgezeigt, die in Zukunft interessant sein könnten. Zur Weiterentwicklung der vorgestellten Methoden und angrenzender Verfahren sollen an dieser Stelle weitere Anregungen gegeben werden. Erweiterungen und Entwicklungsziele, die speziell den 'Karlsruher Endoskopie-Trainer' betreffen, wurden schon am Ende des Kapitels 6 angesprochen.

- Ziel der Modellbildung ist eine möglichst gute Approximation der realen Objekte. Ein statisches Modell der geometrischen Formen ist mit den vorhandenen Methoden abbildbar, die quantitativ korrekte Nachbildung der mechanischen Eigenschaften hingegen ist noch teilweise ungelöst. In Kapitel 3.5 sind zu dieser Thematik schon einige Ansätze aufgezeigt, die jedoch noch ausgebaut werden müssen. Die Topologie des Netzes, die Massenverteilung sowie die Charakteristik und Parameter der Verbindungselemente sollten durch Optimierungsverfahren so angepaßt werden, daß das Modell das reale Verhalten des

Objektes repräsentiert. In dieser Arbeit wurde gemäß der Aufgabenstellung eine qualitativ korrekte Simulation angestrebt und erreicht.

- Sehr wichtig für viele Anwendungen in der 'Virtual Reality' ist neben der visuellen Darstellung auch taktile Information, also das Fühlen einer Kraft bei dem virtuellen Berühren eines Objektes. Die physikalisch-basierten Methoden dieser Arbeit sind hierfür gut geeignet, da die resultierenden Oberflächenkräfte implizit berechnet werden. Probleme bereiten jedoch die der Simulation zugrundeliegenden Diskretisierungen, neben der zeitlichen vor allem die räumliche Diskretisierung infolge des nodalen Netzmodells. Bei Verwendung eines kraftreflektierenden Eingabesystems [MS94] führt dies zu unplausiblen Kraftausgaben und im geschlossenen Regelkreis zu Stabilitätsproblemen. Für entsprechende Problemlösungen ist eine Erweiterung der entwickelten Algorithmen und numerischen Methoden notwendig, insbesondere unter regelungstechnischen Aspekten.

- Die Interaktionsauswertung mit eventueller Objektmodifikation ist abhängig von dem Charakter der Kollision, der Art der kollidierenden Objekte und der Manipulation des Benutzers. Da hierfür nicht nur reine physikalische Gesetzmäßigkeiten modelliert werden können, wurde eine Art Regelbasis eingeführt, die das Verhalten bei Interaktionen spezifiziert (Kapitel 4). Die Implementierung in dieser Arbeit ist jedoch nicht für universelle Szenarien geeignet, sondern hauptsächlich auf die Instrumentarien der Minimal-Invasiven Chirurgie ausgelegt und optimiert. Hier kann ein erweiterter, wissensbasierter Ansatz eine große Hilfe sein, um ein umfassendes Verhaltensmodell aufzustellen und damit eine strukturierte und allgemeingültige Interaktionsbehandlung zu ermöglichen.

- Die Darstellungsform der NURBS besitzt gegenüber der direkten polygonalen Ausgabe gewisse Vorteile (Kapitel 5). Bei zukünftigen Rechnergenerationen wird sich der höhere Bedarf an Rechenleistung für die Tesselation weniger negativ auswirken. Erforderlich ist auch weitere wissenschaftliche Forschung auf dem Gebiet der Freiformflächen bezüglich interaktiver Modifikationen sowie der Strukturveränderung bei Objektmanipulationen. Methoden hierzu wurden schon teilweise entwickelt [Far88], die Komplexität und das Datenaufkommen ist jedoch sehr hoch und noch nicht in Echtzeit zu realisieren, da bei der Modellbildung und Simulation sehr viele Nebenbedingungen einzuhalten wären.

- Für die Genauigkeit und Stabilität der Simulation sind die numerischen Aspekte sehr wichtig. In Kapitel 3.3 wurde hierauf schon ausführlich eingegangen und verschiedene Verfahren erläutert. Zur Steigerung der Effizienz und Güte ist hinsichtlich der Numerik noch einiges Entwicklungspotential vorhanden. Bei der Lösung der Differentialgleichungssysteme sind adaptive Algorithmen vorstellbar, die je nach Objektzustand und Anregung eine Anpassung der Simulation vornehmen.

Anhang A: Tabellen

Tabelle A.1: Objekt-Attribute

Die in dieser Tabelle spezifizierten Objekt-Attribute legen verschiedene Eigenschaften der dynamischen Simulation eines deformierbaren Objektes fest. Bei der Definition wird das Bitvektor-System verwendet, wobei ein Bit des Objekt-Parameters ein bestimmtes Attribut repräsentiert. Dieser Objekt-Parameter kann als ganzes über die Geometrie-Definitionsdatei (Anhang B) oder über Skript-Befehle (Anhang C) gesetzt werden, einzelne Attribute können ebenfalls über einen Skript-Befehl geändert werden. Im allgemeinen ist die jeweilige Option bei gesetztem Bit aktiviert.

Bit / Hex	**Token** (für Skript-Befehl)	**Erläuterung**
4 (0x000010)	ELADYN_MOVE	Globale Bewegung des Objektes, Kraftkopplung auf Vaterknoten freigegeben
5 (0x000020)	ELADYN_POLYDRAW	Darstellung des Objekts als polygonales Dreiecksnetz
6 (0x000040)	ELADYN_GRAV	Schwerkraftvektor für Masseknoten aktiviert, $a_g = (0, g, 0)$ m/s^2
7 (0x000080)	ELADYN_GROUND	Bodenfläche aktiviert ($y = 0$), Positionsbeschränkung der Masseknoten auf $y > 0$
8 (0x000100)	ELADYN_QDYN	Numerik: Auswahl der quasidynamischen Berechnung (resultierende Kraft proportional der Geschwindigkeit)
9 (0x000200)	ELADYN_CALC_TIME	Numerik: Feste Vorgabe der Integrationsschrittweite (Nicht-Echtzeit)
10 (0x000400)	ELADYN_INTER	Verhindert die Interaktionsdetektion bezüglich des Objektes
11 (0x000800)	ELADYN_NURBSDRAW	Darstellung des Objekts als Freiformfläche (NURBS)
12 (0x001000)	ELADYN_SLOWMOT	Numerik: Multiplikation der Integrationsschrittweite mit Zeitfaktor
13 (0x002000)	ELADYN_CALC_NORM	Numerik: Euler-Cauchy-Verfahren, implizite Dämpfung
14 (0x004000)	ELADYN_CALC_KUHN	Numerik: Modifiziertes Euler-Verfahren mit gewichteter Geschwindigkeitsanpassung
15 (0x008000)	ELADYN_TEST	Testflag
16 (0x010000)	ELADYN_NOBEND	Kein automatisches Einfügen von Biegeelementen bei Rohrgeometrien
17 (0x020000)	ELADYN_NORMLOCAL	Normalenberechnung der Knoten erfolgt global (anstatt lokal in Bezug zu Nachbarn)
18 (0x040000)	ELADYN_NORMSHOW	Anzeige der Normalen bei polygonaler Darstellung
19 (0x080000)	ELADYN_DRAWFINE	Verfeinerte Visualisierung bei polygonaler Darstellung
20 (0x100000)	ELADYN_CALC_I	Numerik: Mittelpunkt-Methodik, gemittelte Geschwindigkeit
21 (0x200000)	ELADYN_CALC_II	Numerik: Zweistufenverfahren: Runge-Kutta 2. Ordnung
22 (0x400000)	ELADYN_CALC_III	Numerik: Zweistufenverfahren: Heun-Verfahren
23 (0x800000)	ELA_COAG_SPARK	Flag zur Aktivierung von Koagulationsfunken an interagierenden Knoten

Tabelle A.2: Masseknoten-Attribute

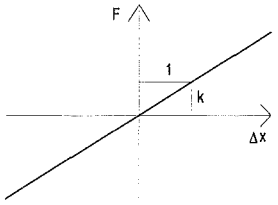
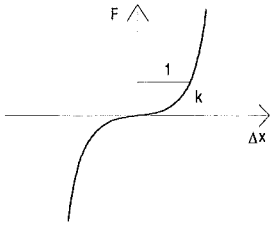
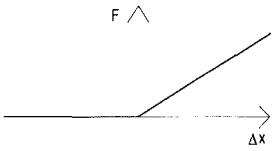
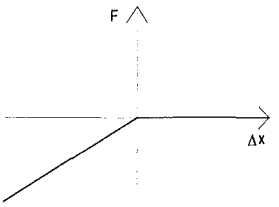
Masseknoten-Attribute definieren verschiedene Zustände und Eigenschaften eines Masseknotens. Es findet ebenfalls das Bitvektor-System Verwendung, wobei ein Bit des Knoten-Parameters einen bestimmten Zustand repräsentiert. Dieser Masseknoten-Parameter kann nur für freie Knoten über die Geometrie-Definitionsdatei (Anhang B) gesetzt werden. Sinnvoll ist dies jedoch nur für die Markierung bestimmter Knoten als nicht interaktiv (ELA_NO_INT) beziehungsweise zur eindeutigen Kennzeichnung hinsichtlich Objektmodifikationen. Andere, objektübergreifende Kopplungs-Attribute können über die 'ELADYN_HYPER'-Datei (Anhang B.4) definiert werden. Die sonstigen Attribut-Bits werden in Abhängigkeit der Objektmanipulationen vom Simulationssystem gesetzt. Im allgemeinen ist die jeweilige Option bei gesetztem Bit aktiviert.

Bit / Hex	Token	Erläuterung
0 (0x000001)	ELA_INT_GRASP	Knoten ist momentan fest gepackt, starre Positionsbindung an Effektor
1 (0x000002)	ELA_INT_CLIP	Knoten ist von einer Klemme gebunden, starre Positionsbindung an Vaterknoten
2 (0x000004)	ELA_INT_CUT	Am Knoten wurde Geometrie lokal/partiell aufgetrennt, Knoten wurde vervielfacht
3 (0x000008)	ELA_INT_DEVID	Am Knoten wurde Geometrie global/total aufgetrennt, Knoten wurde vervielfacht (Modellmodifikation)
4 (0x000010)	ELA_HYP_FIXED	Masseknoten ist fest an globale Position gebunden, räumliche Ortsfixierung
5 (0x000020)	ELA_HYP_MASTER	Knoten ist definiert als Master-Knoten bei Objektverknüpfungen (Positionsbestimmend)
6 (0x000040)	ELA_HYP_SLAVE	Knoten ist definiert als Slave-Knoten bei Objektverknüpfungen (Kraftrückgebend)
7 (0x000080)	ELA_HYP_SPRING	Knoten ist definiert als Verbindungselement-Knoten bei Objektverknüpfungen (Globale VE)
8 (0x000100)	ELA_FOR_INPUT	Knoten ist aktueller Krafteingabeknoten (interaktive Kraftvektorvorgabe über Bedienoberfläche)
9 (0x000200)	ELA_GLO_DYN	Knoten ist Teil eines globaldynamischen Objektes
10 (0x000400)	ELA_FREE_KNOT	F-Knoten, keine Einbindung in geometrische Formen des Objekts
11 (0x000800)	ELA_CUT_MARK	Markierter Knoten für die Verwaltung der Objektmanipulation
12 (0x001000)	ELA_NO_INT	Knoten ist als Interaktions-frei definiert, es findet keine Interaktionsdetektion statt
13 (0x002000)	ELA_INT_COAG	Knoten wird als 'koaguliert' gekennzeichnet
14 (0x...10000)	ELA_MANIP_KNOT	Eindeutige Kennzeichnung von bestimmten Masseknoten Markierung für Objektmodifikation nach Interaktionen

Tabelle A.3: Verbindungselemente

Verbindungselemente (VE) sind kraftübertragende Komponenten, die zwei dynamische Masseknoten miteinander verknüpfen. Ein System von Masseknoten und Verbindungselementen repräsentiert ein elastodynamisches Objekt. Einfachstes Beispiel eines VE ist eine lineare Feder, prinzipiell sind mit der MESD-Methodik (Kapitel 3) jedoch auch beliebige nichtlineare VE möglich. Die Kennlinie der Elemente sollte jedoch aus Stabilitätsgründen stetig sein. Eine schon implementierte Auswahl zeigt Tabelle A.3. Eingangsgrößen eines VE sind die Raumpositionen (oder auch Geschwindigkeiten) der verknüpften Knoten. Ausgangsgrößen sind jeweils dreidimensionale Kraftvektoren, die auf diese Masseknoten wirken. Die Richtung der Vektoren ist im allgemeinen parallel zum Verbindungsvektor der Knoten.

Es findet das Bitvektor-System Verwendung, wobei ein Bit des VE-Parameters einen bestimmte Charakteristik repräsentiert. Diese Eigenschaften sind beliebig kombinierbar, solange sie sich nicht gegenseitig ausschließen. Zur Definition werden ein oder zwei Parameter benötigt: ein Proportionalitätsfaktor k sowie teilweise ein Zusatzparameter z_par. Die Definition dieser Parameter in der Geometriedatei ist im Anhang B.2 sowie B.4 aufgeführt.

Bit/Index/Token	Math. Definition / Beschreibung	Kennlinie
0 / 0x0001 VE_LINEAR	$F = f(\Delta x) = k(\lVert \mathbf{x}_1 - \mathbf{x}_2 \rVert)$ $\Delta x = \lVert \mathbf{x}_1 - \mathbf{x}_2 \rVert$ **Lineares Federelement**, Federkonstante k [N/m] F: Betrag der resultierenden VE-Kraft [N] $\mathbf{x}_1$: Positionsvektor des ersten Masseknotens $\mathbf{x}_2$: Positionsvektor des zweiten Masseknotens Δx: Abstandsbetrag der Knoten [m]	
1 / 0x0002 VE_QUADRATIC	$F = f(\Delta x) = \begin{cases} +k(\lVert \mathbf{x}_1 - \mathbf{x}_2 \rVert)^2 & \text{für } \lVert \mathbf{x}_1 - \mathbf{x}_2 \rVert \geq 0 \\ -k(\lVert \mathbf{x}_1 - \mathbf{x}_2 \rVert)^2 & \text{für } \lVert \mathbf{x}_1 - \mathbf{x}_2 \rVert < 0 \end{cases}$ **Quadratisches Federelement** Federkonstante k [N/m^2]	
4 / 0x0010 VE_PULL_ONLY	$F = \begin{cases} 0 & \text{für } \Delta x \leq 0 \\ f(\Delta x) & \text{für } \Delta x > 0 \end{cases}$ Beschränkung auf Zugkräfte	
5 / 0x0020 VE_PUSH_ONLY	$F = \begin{cases} f(\Delta x) & \text{für } \Delta x \leq 0 \\ 0 & \text{für } \Delta x > 0 \end{cases}$ Beschränkung auf Druckkräfte	

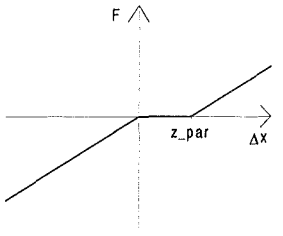
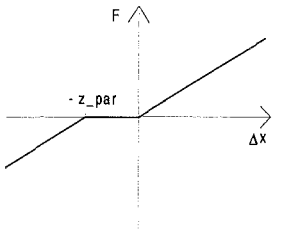
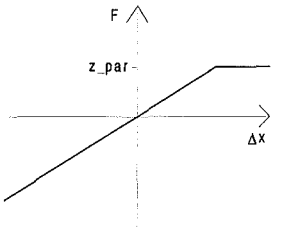
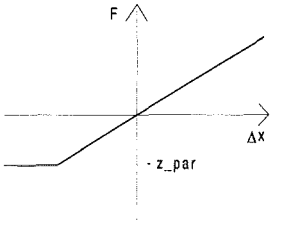
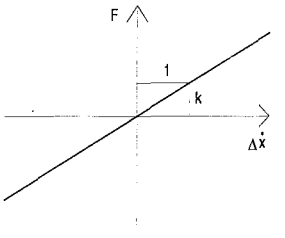
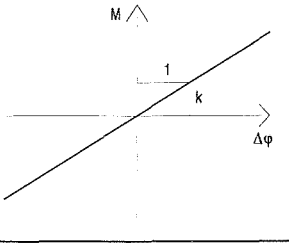
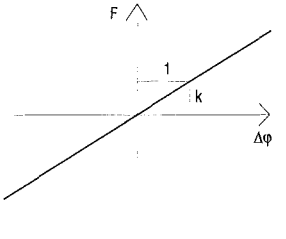
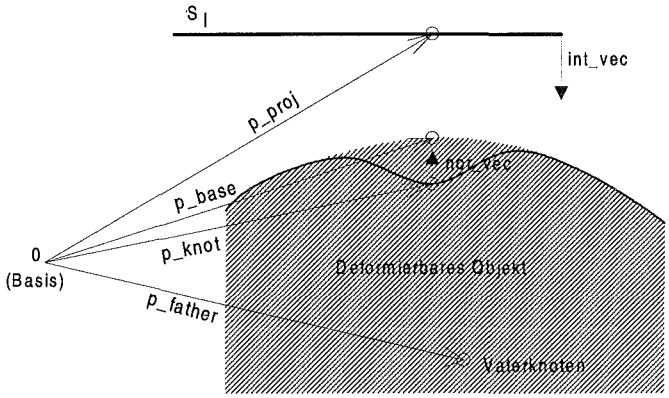
Bit/Index/Token	Math. Definition / Beschreibung	Kennlinie
6 / 0x0040 VE_DEADZONE_P	$F = \begin{cases} f(\Delta x) & \text{für } \Delta x \le 0 \\ 0 & \text{für } 0 < \Delta x \le z\_par \\ f(\Delta x - z\_par) & \text{für } \Delta x > z\_par \end{cases}$ Totzone bei Zugkraft Schrankenparameter *z_par* [m]	
7 / 0x0080 VE_DEADZONE_N	$F = \begin{cases} f(\Delta x + z\_par) & \text{für } \Delta x \le -z\_par \\ 0 & \text{für } -z\_par < \Delta x \le 0 \\ f(\Delta x) & \text{für } \Delta x > 0 \end{cases}$ Totzone bei Druckkraft Schrankenparameter *z_par* [m]	
12 / 0x1000 VE_LIMITED_P	$F = \begin{cases} f(\Delta x) & \text{für } f(\Delta x) \le z\_par \\ f(z\_par) & \text{für } f(\Delta x) > z\_par \end{cases}$ Kraftschranke bei Zugkraft Schrankenparameter *z_par* [N]	
13 / 0x2000 VE_LIMITED_N	$F = \begin{cases} f(-z\_par) & \text{für } f(\Delta x) \le -z\_par \\ f(\Delta x) & \text{für } f(\Delta x) > -z\_par \end{cases}$ Kraftschranke bei Druckkraft Schrankenparameter z_par [N]	
8 / 0x0100 VE_VEL_FUNC	$F = f(\Delta\dot{x}) = k(\lVert\dot{\mathbf{x}}_1 - \dot{\mathbf{x}}_2\rVert)$ $\Delta\dot{x} = \lVert\dot{\mathbf{x}}_1 - \dot{\mathbf{x}}_2\rVert$ **Geschwindigkeitsproportionales VE** (Dämpfungs-, Reibungselement) Proportionalitätsfaktor *k* [Ns/m] $\dot{x}_1$: Geschwindigkeitsvektor des ersten Knotens $\dot{x}_2$: Geschwindigkeitsvektor des zweiten Knotens $\Delta\dot{x}$: Geschwindigkeitsdifferenz der Knoten [m/s]	
9 / 0x0200 VE_TORSION	$M = f(\Delta\varphi) = k(\lVert\varphi - \varphi_0\rVert)$ **Torsionsfederelement** (Zentralknoten) Proportionalitätsfaktor *k* [Nm/rad] M: Resultierendes Drehmoment [Nm] φ: Räumliche Orientierung des Knotens φ_0: Ausgangsorientierung des Knotens	
10 / 0x0400 VE_BEND_EL	$F = f(\Delta\varphi) = k(\varphi_B - \varphi_0)$ $\varphi_B = \angle((\mathbf{x} - \mathbf{x}_1),(\mathbf{x} - \mathbf{x}_2))$ **Biegefederelement** φ_B: Winkel zwischen den jeweiligen Positionsvektoren zu zwei Nachbarknoten φ_0: Ausgangsorientierung des Biegefederelements	

Tabelle A.4: Vektoren der Plausibilitäts-Prüfung

Wie in Kapitel 4 ausführlich erläutert, wird als letzte Stufe der Kollisionsdetektion eine Überprüfung durchgeführt, ob eine Interaktion als 'sinnvoll' zu bezeichnen ist. Durch die zeitliche Diskretisierung der Simulation und örtliche Diskretisierung des deformierbaren Objektes ergeben sich bei ausschließlicher Verwendung geometrischer Tests unplausible Ergebnisse. Die Art und Parameter dieser 'Plausibilitäts-Prüfung' lassen sich an spezifische Erfordernisse anpassen (Tabelle A.5, Anhang C.1). Verschiedene Bedingungen werden kombiniert und logisch 'UND'-verknüpft, um eine möglichst gute Aussage bezüglich der Interaktionswahrscheinlichkeit treffen zu können.

Die Bedingungen bestehen im wesentlichen aus einer Überprüfung zweier Vektoren. Deren Skalarprodukt wird mit einer vorgegebenen Schranke verglichen, hierbei erfolgt der Übergang auf binäre Zustandsgrößen. Bei der Definition der Bedingungen wird das Bitvektor-System verwendet, wobei ein Bit des Typ-Parameters einen bestimmten durchzuführenden Test repräsentiert. Sind alle einzelnen Prüfungen erfolgreich, so ist auch der Plausibilitätstest im ganzen erfolgreich - eine Interaktion ist sinnvoll. Weiterhin ist neben der logischen Verknüpfung der Einzelbedingungen auch eine kombinierte Prüfung möglich, zusätzlich läßt sich die nötige Berechnung der Normalenvektoren variieren.



S_I	Interaktionsstrecke
int_vec	Interaktionswirkungsvektor
nor_vec	Normalenvektor des int. Knoten
p_knot	Ortsvektor des int. Knoten
p_base	Ortsvektor der Ausgangslage des int. Knoten
p_proj	Ortsvektor des auf die Interaktionsstrecke projezierten Knotens
p_father	Ortsvektor des zugehörigen Vaterknotens

$$\text{pb\_val} = \left(\frac{\mathbf{v1}}{\|\mathbf{v1}\|} \cdot \frac{\mathbf{v2}}{\|\mathbf{v2}\|} + 1.0\right) \cdot 0.5 \tag{A.1}$$

$$\text{plaus\_bool} = \begin{cases} \text{TRUE} & \textit{für } \text{pb\_val} \geq \textit{thr\_val} \\ \text{FALSE} & \textit{für } \text{pb\_val} < \textit{thr\_val} \end{cases} \tag{A.2}$$

pb_val: Normierter Wahrscheinlichkeitswert bezüglich Richtungsübereinstimmung der Vektoren **v1** und **v2**, pb_val ∈ [0, 1]

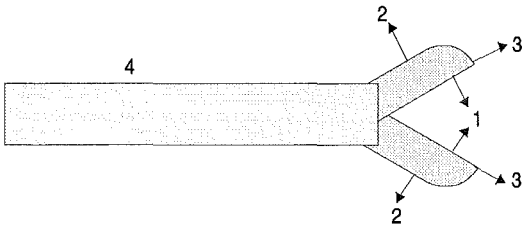
thr_val: Schwellwert der Plausibilitätsprüfung

plaus_bool: Boolsche Variable, repräsentiert das Prüfungsergebnis eines Einzeltests

Bit/Index	Token	Erläuterung
0 / 0x0001	DIR_TEST	v1 = nor_vec v2 = p_base - p_proj Genereller Oberflächentest: Innen/Außen
1 / 0x0002	SURF_TEST	v1 = nor_vec v2 = p_proj - p_father Erweiterter Oberflächentest: Konvexe Geometrie
2 / 0x0004	PICK_TEST	v1 = - nor_vec v2 = int_vec Interaktionswirkungstest
3 / 0x0008	EFF_TEST	v1 = int_vec v2 = p_proj - p_knot Interaktionsrichtungstest
4 / 0x0010	POS_TEST	v1 = nor_vec v2 = p_knot - p_proj Positionswirkungstest
5 / 0x0020	EFF2_TEST	v1 = int_vec v2 = p_knot - p_proj Inverser Interaktionsrichtungstest
8 / 0x0100	COMB_EVAL	Kombinationsprüfung: Alle aktivierten Bedingungen werden gemeinsam überprüft und insgesamt ausgewertet (Produkt aller Einzelwerte ⇔ Produkt aller zugehörigen Schwellwerte)
9 / 0x0200	NORM_MID	Berechnung des Normalenvektors des interagierenden Knotens sowohl aus den Positionen der Nachbarknoten (normal) als auch aus aktueller Knotenposition
10 / 0x0400	NORM_PROJ	Berechnung des Normalenvektors des interagierenden Knotens sowohl aus den Positionen der Nachbarknoten (normal) als auch aus aktueller, auf Interaktionsstrecke projezierter Knotenposition

Tabelle A.5: Token der Plausibilitäts-Prüfung

Die zur Interaktionsdetektion notwendige Plausibilitätsprüfung besitzt für verschiedene Teile einer interagierenden Geometrie (Instrument) auch verschiedene Überprüfungsparameter, um eine Anpassung an spezifische Gegebenheiten des Szenarios und der Modelle zu ermöglichen. Durch ein entsprechendes Skript-Kommando (Anhang C.1) können die Typ-Parameter und die Schwellwerte modifiziert werden. Hierzu wird zuerst ein Token übergeben, welches die Lokalität und den Zustand der Interaktion spezifiziert. Der übergebene Typ-Parameter charakterisiert hierbei die Art der Plausibilitätsprüfung, wie in Tabelle A.4 angeführt.



Beispiel einer interagierenden Geometrie (Instrument) mit den Interaktionsregionen (1-4) und den entsprechenden Interaktions-Wirkungsvektoren.

Token	Bemerkungen
EFFECTOR_I	Interaktionslokalisation: Effektor, Innenseite - Knoten noch nicht interagierend (1)
EFFECTOR_IO	Interaktionslokalisation: Effektor, Innenseite - Knoten schon interagierend (1)
EFFECTOR_A	Interaktionslokalisation: Effektor, Außenseite - Knoten noch nicht interagierend (2)
EFFECTOR_AO	Interaktionslokalisation: Effektor, Außenseite - Knoten schon interagierend (2)
EFFECTOR_T	Interaktionslokalisation: Effektor - Test auf Innen/Außen-Interaktion
SHAFT	Interaktionslokalisation: Schaft - Knoten noch nicht interagierend (4)
SHAFT_O	Interaktionslokalisation: Schaft - Knoten schon interagierend (4)
HEAD	Interaktionslokalisation: Effektorspitze - Knoten noch nicht interagierend (3)
HEAD_O	Interaktionslokalisation: Effektorspitze - Knoten schon interagierend (3)
HEAD_Z	Interaktionslokalisation: Effektorspitze, Zusatztest - Knoten noch nicht interagierend (3)

Die folgenden Plausibilitäts-Token haben Ausschlußcharakter, der übergebene Typ-Parameter definiert durch gesetzte Bits diejenigen Prüfungen, die nicht durchgeführt werden. Auf diese Weise kann die Plausibilitätsprüfung in Abhängigkeit des interagierenden Objektprimitivs weiter spezifiziert werden. Der übergebene Schwellwert wird mit den weiteren definierten Schwellwerten multipliziert. Daraus ergibt sich dann die endgültige Plausibilitätsschwelle, die als Grundlage für die Interaktionsentscheidung dient.

Token	Bemerkungen
ORGAN	Interaktionsobjekt: Zweifach geschlossene Geometrie ('Organ', object_type = 1)
PIPE	Interaktionsobjekt: Einfach geschlossene Geometrie ('Rohr', object_type = 2)
FLAT	Interaktionsobjekt: Offene Geometrie ('Gewebe', object_type = 3)
EXPLICIT	Interaktionsobjekt: Explizit definierte Geometrie (object_type = 5)

Anhang B: Spezifikation der Objekte

Die geometrischen und elastodynamischen Daten der deformierbaren Objekte und der übergeordneten Beziehungen werden in ASCII-Dateien abgelegt, die der KISMET-spezifischen Syntax entsprechen. An dieser Stelle werden nur die für diese Arbeit besonders relevanten Primitive und Definitionen aufgeführt. Weitere Spezifikationen sind dem KISMET-Leitfaden [KIS93] zu entnehmen, insbesondere die Definitionen zu Simulations-organisation, Licht- und Materialeigenschaften, Kinematik und 'starren' Geometrien. Prinzipiell sind die wichtigen Parameter der Elastodynamik auch zur Simulationslaufzeit über entsprechende Menüs der Bedieneroberfläche oder auch über Skript-Befehle (Anhang C) modifizierbar.

B.1 Allgemeine Definitionen

B.1.1 Notation

Die Spezifikation einer Datenstruktur wird in hierarchische Definitionsblöcke zerlegt, darunter folgt eine erläuternde Parameter-Beschreibung. Zur Definition der Modelle in KISMET wird folgende Notation verwendet:

< ... >	variable Definitionselemente (non-terminals)
::=	Substitutionsanweisung
[...] • *k*	Elemente zwischen [ ] werden *k* mal wiederholt
|	Trennung von Definitionsalternativen
{ ... }	wahlfreie Definition, Elemente zwischen { } dürfen entfallen
/* ... */	Kommentarklammer, der Text dient zur Erläuterung der Syntaxdefinition
" ... "	Die Zeichenkette zwischen " " ist Definitionskonstante (terminals)
if (...) { ... }	Bedingte Definition. { } ist nur relevant, wenn Bedingung in ( ) gültig ist
for (all *k*) {..}	Definitionsschleife. { } wird für alle *k* durchlaufen

Weiterhin werden als Variablentypen verwendet:

float	Zahl in Fließkommaformat oder wissenschaftlicher Notation (E-Format), Wortbreite 32 Bit
integer	Ganze Zahl, Wortbreite 32 Bit
unsigned	Natürliche ganze Zahl (≥ 0)
char	ASCII-Zeichen oder ganze Zahl, Wortbreite 8 Bit
string	ASCII-Zeichenkette, Feld von char-Werten (char[..])
(k1 ... k2)	Der zulässige Wertebereich des Elements liegt im Intervall $k \in [k1, k2]$
(>= k1)	Der zulässige Wertebereich des Elements beginnt bei k1

B.1.2 Prinzipieller Syntax der KISMET Geometrie-Definitionen

```
<geo_file> = KISMETGEO ::=
  <comment>          ::= string
  <GEO_Body>         ::= POLYHEDRON | 3D_SOLID_DATASET
END KISMETGEO

3D_SOLID_DATASET ::=
  <solid_id>              ::= "0"
  <geo_body>              ::= 3D_SOLID_PRIMITIVE
 < material_index>        ::= char (1...255)
END 3D_SOLID_DATASET

3D_SOLID_PRIMITIVE        := SPHERE | BOX | CONE | CYLINDER | SWEEP |
                             CURVE | SURF | .........
                              | NURBS | ELADYN_NURBS | ELADYN_POLY
```

Syntax der Geometrie-Datei Spezifikation in KISMET ('.mpo'-Datei)

comment Kommentar, Textuelle Charakterisierung bezüglich des geometrischen Objektes (wird als String gelesen, muß mindestens ein Zeichen und darf keine Leerzeichen enthalten)

solid_id zur Unterscheidung zwischen POLYHEDRON-Primitiv und anderen Geometrie-Primitiven

geo_body eigentliche Geometrie-Definition. Je nach Art des Primitivs erfolgt die genaue Spezifikation in unterschiedlichen Blöcken. Beispielsweise sind Definitionen in Form polygonaler Darstellungen, Freiformflächen und auch als CSG-Objekte[1] möglich.

material_index 'Material'-Attribut des definierten Objektes. Der Index bezieht sich auf die Material-Tabelle, die in der Material-Datei spezifiziert ist. Hiermit können dem Objekt neben den Farbwerten auch diverse Lichtreflexionswerte zugewiesen sowie Texturen zugeordnet werden.

B.2 Syntax des ELADYN_NURBS - Primitives

Das ELADYN_NURBS-Primitiv verknüft die geometrische Ausprägung der Freiformfläche (NURBS)[2] mit der unterlagerten, elastodynamischen Netzstruktur. Wegen des Grundsatzes der größtmöglichen Trennung von geometrischen und elastodynamischen Daten wird das Primitiv aus zwei getrennten Teilblöcken gebildet.

```
ELADYN_NURBS ::=
  <id>              ::= "90"
  <nurbs_def>       ::= NURBS_OBJ
  <eladyn_def>      ::= ELADYN_OBJ
END ELADYN_NURBS
```

Spezifikation des ELADYN_NURBS-Primitives

[1] Constructive Solid Geometry: Geometrische Grundkörper (Quader, Kugel, Zylinder, Kegel usw.), mit denen sich über boolsche Operationen komplexere Geometrien definieren lassen.
[2] Non-Uniform Rational B-Spline Surface

B.2.1 Syntax des NURBS_OBJ - Primitives

```
NURBS_OBJ ::=
  <surf_count>     ::= unsigned (>= 1)
  <surf_list>      ::= surf_count • [<NURBS_SURF>]

  <bind_count>     ::= unsigned (>= 0)
  <bind_list>      ::= bind_count • [<NURBS_BINDINGS>]
END NURBS_OBJ

NURBS_BINDINGS ::=
  <bind_surf>      ::= unsigned (>= 0), (<= <surf_count>)
  <bind_cp>        ::= unsigned (>= 0)
  <bind_surf_to>   ::= unsigned (>= 0), (<= <surf_count>)
  <bind_cp_to>     ::= unsigned (>= 0)
END NURBS_BINDINGS
```

Spezifikation des NURBS_OBJ-Primitives

surf_count Anzahl der verschiedenen NURBS-Oberflächen, die das NURBS-Objekt bilden (Patches). Ein NURBS-Objekt kann aus mehreren verschiedenen, unabhängig definierten Oberflächen bestehen.

bind_count Anzahl der Bindungen zwischen Kontrollpunkten. Ein Kontrollpunkt einer Oberfläche kann an einem Kontrollpunkt einer anderen Oberfläche fixiert werden. Auf diese Weise können auch Oberflächen mit sich ändernder Form (deformierbare Objekte) ein geschlossenes Objekt bilden.

bind_surf Nummer der Oberfläche des ersten (zu bindenden) Kontrollpunktes

bind_cp Nummer des zu bindenden Kontrollpunktes der selektierten Oberfläche

bind_surf_to Nummer der Oberfläche des zweiten (freien) Kontrollpunktes

bind_cp_to Nummer des Kontrollpunktes der selektierten Oberfläche, an dem der erste Kontrollpunkt fixiert wird

```
NURBS_SURF ::=
  <s_type>              ::= "1" | "2" | "3" | "4"
  <t_ord>               ::= unsigned (>= 1)
  <n_t_knots>           ::= unsigned
  for (all n_t_knots)
    <t_knots_list>      ::= <t_knot_par> • n_t_knots
  <s_ord>               ::= unsigned (>= 1)
  <n_s_knots>           ::= unsigned
  for (all n_s_knots)
    <s_knots_list>      ::= <s_knot_par> • n_s_knots
  <scaling>             ::= float (> 0)
  <trans>               ::= <trans_par>
  <trans_mat>           ::= <trans_mat_par>
  if ( s_type >= 3 )  {
    <s_closed>          ::= unsigned (>= 0)
    <t_closed>          ::= unsigned (>= 0)
  }

  for ( all (n_s_knots - s_ord - s_closed) )
    for ( all (n_t_knots - t_ord - t_closed) )
      <cpoint_mat>      ::= <cp_coord>
```

```
  <n_trims>              ::= unsigned (>= 0)
    <trim_body>          ::= <NURBS_TRIM>  • n_trims

  <tex_type>             ::= "0" | "1" | "2"
  if (tex_type > 0 )
    <tex_body>           ::= <NURBS_TEX>
END NURB_SURF

<t_knot_par>             ::= float
<s_knot_par>             ::= float
<trans_par>              ::= <x>, <y>, <z>
<trans_mat_par>          ::= float[4][4]
<cp_coord>         ::=
  if (s_type == 1 | 3)  <cp_coord> ::= <cp_coord_n>
  else                  <cp_coord> ::= <cp_coord_r>
<cp_coord_n>             ::= <x>, <y>, <z>
<cp_coord_r>             ::= <x>, <y>, <z>, <w>
<x>,<y>,<z>,<w>          ::= float
```

Spezifikation eines NURBS-Primitives

s_type Typ der NURBS-Fläche. Definiert den Typ der verwendeten B-Splines (rational / nichtrational) und den Flächentyp. Erweiterte Oberflächen werden zum Erzeugen geschlossener Geometrien verwendet, hierzu werden die Randkontrollpunkte gespiegelt und wiederholt.
1: Nichtrationale B-Splines
2: Rationale B-Splines
3: Nichtrationale B-Splines, Erweiterte Oberfläche
4: Rationale B-Splines, Erweiterte Oberfläche

t_ord Ordnung der verwendeten B-Splines in t-Richtung

n_t_knots Anzahl der Spline-Knoten in t-Richtung

t_knots_list Liste der B-Spline-Knoten in t-Richtung (aufsteigende Ordnung)

s_ord Ordnung der verwendeten B-Splines in s-Richtung

n_s_knots Anzahl der Spline-Knoten in s-Richtung

s_knots_list Ordnung der verwendeten B-Splines in s-Richtung

scaling Skalierungsfaktor. Alle Kontrollpunkte werden mit diesem Faktor multipliziert, daraus ergeben sich die kartesischen Koordinaten der Kontrollpunktpositionen.

trans Parameter zur Verschiebungs-Transformation in x-, y- und z-Richtung. Alle Kontrollpunkte werden mit diesen Werten transformiert, daraus ergeben sich die kartesischen Koordinaten der Kontrollpunktpositionen.

trans_mat Transformationsmatrix (4 x 4). Alle Kontrollpunkte werden mit dieser Matrix transformiert, daraus ergeben sich die kartesischen Koordinaten der Kontrollpunktpositionen.

s_closed Wiederholungsfaktor in s-Richtung, notwendig zur Erzeugung geschlossener Oberflächen. Je größer der Wert, desto größer ist der Grad der Stetigkeit an den anstoßenden Randkurven.

t_closed Wiederholungsfaktor in t-Richtung

cp_coord_n Position des Kontrollpunktes in kartesischen Koordinaten
Version: nichtrational, ohne Wertigkeit

cp_coord_r Position des Kontrollpunktes in kartesischen Koordinaten
Version: rational, mit Wertigkeit (homogene Koordinaten)

n_trims Anzahl der definierten Trimming-Kurven. Eine Trimming-Kurve definiert den sichtbaren Bereich eines NURBS.

tex_type Typ der Textur-Projektion
0: Keine Textur-Projektion
1: Textur-Projektion mit nichtrationalen Parametern
2: Textur-Projektion mit rationalen Parametern

```
NURBS_TRIM  ::=
  <trim_type>                  ::=  "1" | "2" | "3" | "4"
  if (trim_type < 3)  {
    <trim_ord>                 ::=  unsigned
    <knot_cnt>                 ::=  unsigned
    for (all knot_cnt)
      <trim_knots>             ::=  float
    for (all (knot_cnt - trim_ord))
      <trim_cp_mat>            ::=  <trim_point>
  }
  else  {
    <pnt_cnt>                  ::=  unsigned
    for (all pnt_cnt )
      <trim_pnt_mat>           ::=  <trim_point>
  }
  <connect> ::=  "0" | "1"
END NURBS_TRIM

<trim_point>  ::=
  if (trim_type == 1, 3)
    <trim_point>               ::=  <trim_point_n>
  else
    <trim_point>               ::=  <trim_point_r>
  <trim_point_n>               ::=  <s>, <t>
  <trim_point_r>               ::=  <s>, <t>, <w>
 <s>, <t>, <w>                 ::=  float
```

Spezifikation einer NURBS_TRIM-Struktur (Trimming-Kurven)

trim_type **Typ der Trimming-Kurve**
1: Spline-basierte Kurve, nichtrationale Parameter
2: Spline-basierte Kurve, rationale Parameter
3: Stückweise lineare Kurve, nichtrationale Parameter
4: Stückweise lineare Kurve, rationale Parameter

trim_ord Ordnung der Spline-basierten Kurve

knot_cnt Anzahl der Spline-Knoten

trim_knots Liste der Trimming-Knoten (aufsteigende Ordnung)

pnt_cnt Anzahl der Punkte (Stückweise lineare Kurve)

trim_cp_mat Positionsmatrix der Trimming-Punkte (Spline-basierte Kurve)

trim_pnt_mat Positionsmatrix der Trimming-Punkte (Stückweise lineare Kurve)

connect Index, der die definierte Kurve mit der vorher definierten verbindet. Damit ist die Erzeugung komplexer Trimming-Kurven möglich, die aus mehreren stückweise definierten Kurven bestehen.

trim_point_n Positionsparameter des Trimming-Punktes (nichtrational)

trim_point_r Positionsparameter des Trimming-Punktes (rational)

```
NURBS_TEX  ::=
  <tex_t_ord>           ::=  unsigned
  <n_tt_knots>          ::=  unsigned
  for (all n_tt_knots)
    <tex_t_knots>       ::=  float
  <tex_s_ord>           ::=  unsigned
  <n_ts_knots>          ::=  unsigned
  for (all n_ts_knots)
    <tex_s_knots>       ::=  float
  for ( all (n_ts_knots - tex_s_ord) )
    for ( all (n_tt_knots - tex_t_ord) )
      <tex_point_mat>   ::= <tex_point>
END NURBS_TEX

<tex_point>  ::=
  if (tex_type == 1)
    <tex_point>         ::=  <tex_point_n>
  if (tex_type == 2)
    <tex_point>         ::=  <tex_point_r>
  <tex_point_n>         ::=  <s>, <t>
  <tex_point_r>         ::=  <s>, <t>, <w>
 <s>, <t>, <w>          ::=  float
```

Spezifikation der NURBS_TEX - Struktur (Texturprojektion, 'Texture Mapping')

tex_t_ord Ordnung der Textur-Projektionskurve in t-Richtung

n_tt_knots Anzahl der Knoten der Textur-Projektionskurve in t-Richtung

tex_t_knots Liste aller Knoten der Textur-Projektionskurve in t-Richtung (aufsteigende Ordnung)

tex_s_ord Ordnung der Textur-Projektionskurve in s-Richtung

n_ts_knots Anzahl der Knoten der Textur-Projektionskurve in s-Richtung

tex_s_knots Liste aller Knoten der Textur-Projektionskurve in s-Richtung (aufsteigende Ordnung)

tex_point_mat Matrix der Texturprojektions-Kontrollpunkte

tex_point_n Parameter des Texturprojektions-Kontrollpunkts (nichtrational)

tex_point_r Parameter des Texturprojektions-Kontrollpunkts (rational)

B.2.2 Syntax des ELADYN_OBJ - Primitives

```
ELADYN_OBJ ::=
  <eladyn_type>   ::= ("1" | "2" | "3" | "4" ) + <version>
    <prop_def>    ::= ELADYN_PROP
    <knot_def>    ::= ELADYN_KNOT
    <spring_def>  ::= ELADYN_SPRING
END ELADYN_OBJ

<version>   ::= "300"      /* Neueste Version ! */
```

Spezifikation des ELADYN_OBJ - Primitives

```
ELADYN_PROP       :=
  <object_type>   ::= unsigned
  <knot_num_t>    ::= unsigned
  <knot_num_s>    ::= unsigned
  if (ELADYN_POLY)  {              /* Nur bei ELADYN_POLY - Primitiv ! */
    tex_max_t     ::= float
    tex_max_s     ::= float
  }
  <obj_flags>     ::= unsigned
  <k_mass>        ::= float
  <k_damp>        ::= float
  <sconst_geo>    ::= float
  <obj_mass>      ::= float
  <sconst_ret>    ::= float
  <obj_damp>      ::= float
  <sconst_obj>    ::= float
  <sconst_tor>    ::= float
  <inert_type>    ::= unsigned
  if (inert_type > 1)   {
    <inert_x>     ::= float
    <inert_y>     ::= float
    <inert_z>     ::= float
  }
  <tear_force>    ::= float
  <qdyn_fact>     ::= float
  <fine_fact>     ::= float
  <mult_fact>     ::= float
  <bend_param>    ::= float
  <twist_param>   ::= float
  <ela_param>     ::= float
  <inter_val>     ::= float
  <len_adapt>     ::= float
  <len _adapt2>   ::= float
  <inter_param>   ::= float
  <force_f1>      ::= float
  <force_f2>      ::= float
  <force_f3>      ::= float
END ELADYN_PROP
```

Spezifikation der ELADYN_PROP - Struktur

eladyn_type Typ des elastodynamischen Objektes
1: Keine Knotenbeschränkungen, nur G-Knoten
2: Knotenbeschränkungen definierbar, nur G-Knoten
3: Keine Knotenbeschränkungen, freie Knoten definierbar
4: Knotenbeschränkungen definierbar, freie Knoten definierbar
+ 100 : Version 1 - Objekterweiterung (ab KISMET V.4.95.x)
+ 200 : Version 2 - Erweiterte Parameterliste (ab KISMET V.4.95.4)
+ 300 : Version 3 - Erweiterte Funktionalität (ab KISMET V.4.96.x)

object_type Objektspezifikation, definiert Objektprimitiv
1: Geschlossenes Objekt ('ORGAN'), in beiden Oberflächenricht. geschl.
2: Halbgeschlossenes Objekt ('PIPE'), in einer Oberflächenrichtung geschl.
3: Offenes Objekt ('FLAT'), in keiner Oberflächenrichtung geschlossen
5: Explizite Definition des Objektes
9: Nicht näher spezifiziertes Objekt

knot_num_t Anzahl der G-Knoten in t-Richtung bei regelmäßiger Netztopologie

knot_num_s Anzahl der G-Knoten in s-Richtung bei regelmäßiger Netztopologie

tex_max_t Maximaler Faktor der Texturparameter in t-Richtung
Wird zur automatischen Generierung der Texturparameter bei dem ELADYN-POLY-Primitiv benötigt, beim ELADYN_NURBS-Primitiv erfolgt die Texturparameterdefinition innerhalb der NURBS-Definition

tex_max_s Maximaler Faktor der Texturparameter in s-Richtung

obj_flags Definiert die Atribute des elastodynamischen Objektes
⇒ siehe Tabelle A.1

k_mass Masse der Knoten mit direktem Geometriebezug (G-Knoten) [kg]

k_damp Dämpfungsfaktor der G-Knoten (implizite Dämpfung) [N·s/m]
Die implizite, viskose Dämpfungskraft ist proportional zum Dämpfungsfaktor und der Geschwindigkeit des entsprechenden Knotens

sconst_geo Elementkonstante der Standard-Federelemente (verbindet zwei G-Knoten)
Zwei Federcharakteristiken sind möglich:
sconst > 0 : Federkraft proportional zur Federauslenkung [N/m]
sconst < 0 : Federkraft prop. zur quadratischen Federauslenkung [N/m²]

obj_mass Masse der inneren Zentralknoten [kg]

sconst_ret Elementkonstante der Ruhelagenfederelemente (Verformungskraft)
Zwei Federcharakteristiken sind möglich:
sconst > 0 : Federkraft proportional zur Federauslenkung [N/m]
sconst < 0 : Federkraft prop. zur quadratischen Federauslenkung [N/m²]

obj_damp Dämpfungsfaktor der Zentralknoten (implizite Dämpfung) [N·s/m]
Bestimmt die viskose Dämpfung der globalen Knotenbewegungen

sconst_obj Standardwert der Z-Elementkonstante (Verbindung mit mindestens einem inneren Z-Knoten). Zwei Elementcharakteristiken sind möglich:
sconst > 0 : Federkraft proportional zur Federauslenkung [N/m]
sconst < 0 : Federkraft prop. zur quadratischen Federauslenkung [N/m²]

sconst_tor Elementkonstante der Torsionsfederelemente an Zentralknoten [N/rad] (bezüglich der globalen Orientierung der Zentralknoten)

inert_type Definiert die Art der Rotationsträgheit
(bezüglich der globalen Orientierung der Zentralknoten)
0: Keine Rotationsträgheit
1: automatische Berechnung der Trägheitsparameter (aus Bounding-Box und Massen)
2: explizite Angabe der drei Haupt-Trägheitsparameter

inert_x Rotations-Trägheitsparameter in Richtung der x-Hauptachse [kg·m²]

inert_z Rotations-Trägheitsparameter in Richtung der y-Hauptachse [kg·m²]

inert_z Rotations-Trägheitsparameter in Richtung der z-Hauptachse [kg·m²]

tear_force Zerreißkraft eines Masseknotens [N]; bei Überschreitung der Interaktionskraft an einem Masseknoten erfolgt vordefinierte Modellmodifikation

qdyn_fact Geschwindigkeits-Proportionalitätsfaktor bei quasidynamischer Rechnung (Numerische Auswertung der Elastodynamik, $\in [0,1]$)

fine_fact Proportionalitäts-Faktor bei der Generierung von Zwischenpunkten in polygonaler Darstellung, $\in [0,1]$)

mult_fact Proportionalitäts-Faktor bei der Speicherallokierung von Elementen des elastodynamischen Modells (> 1.0). Ein großer Wert erhöht den Bedarf an Speicherplatz; größere Modellmodifikationen sind aber somit ohne Neuallokierung von Speicher möglich

bend_param Elementkonstante eines Biegefederelementes [N/rad] zwischen zwei Zentralknoten

twist_param Elementkonstante eines Torsionsfederelementes [N/rad] zwischen einem Zentralknoten und der Geometriehauptachse

ela_param Freier Parameter

inter_val Interaktions-Faktor: Schwellwert bei der Plausibilitätsprüfung ($\in [0,1]$)

len_adapt Faktor zur Anpassung der Verbindungselemente bei Modellmodifikationen. Kürzung der Verbindungselemente beim Schneiden ($\in [0,1]$)

len_adapt2 Faktor zur Anpassung der Verbindungselemente bei Modellmodifikationen Kürzung der Verbindungselemente beim Koagulieren ($\in [0,1]$)

inter_param Freier Parameter

force_f1
force_f2
force_f3 Parameter für Kraftreflektion

```
ELADYN_KNOT ::=
  <n_knots>              ::= unsigned
  if (eladyn_type > 2)
    <geo_knots>          ::= unsigned
  else
    <geo_knots>          ::= <n_knots>
  for(all geo_knots) {                          /* Alle Geo-Knoten */
    <kindex>             ::= int
    if (eladyn_type == 2,4) {
      <bound_x>          ::= "0" | "1"
      <bound_y>          ::= "0" | "1"
      <bound_z>          ::= "0" | "1"
      <bound_all>        ::= "0" | "1"
    }
  }
  for(all (n_knots - geo_knots)) {              /* Alle 'freie' Knoten */
    <knot_coord>         ::= <x>, <y>, <z>
    if (eladyn_type == 4) {
      <bound_x>          ::= "0" | "1"
      <bound_y>          ::= "0" | "1"
      <bound_z>          ::= "0" | "1"
      <bound_all>        ::= "0" | "1"
    }
    <knot_type>          ::= unsigned
    <knot_mass>          ::= float
    <father_knot>        ::= unsigned
  }
END ELADYN_KNOT

<x>, <y>, <z>            ::= float
```

Spezifikation der ELADYN_KNOT - Struktur

n_knots Gesamtanzahl aller Masseknoten des elastodynamischen Netzes

geo_knots Anzahl der Masseknoten mit direktem Geometriebezug (G-Knoten, geo_knots <= n_knots). Die Differenz fz_knots = n_knots - geo_knots ist die Anzahl der Masseknoten ohne direkten Geometriebezug (F-Knoten + Z-Knoten).

k_index Index des entsprechenden Konrollpunktes im NURBS Konrollpunkt-Array. Die Position dieses Kontollpunktes wird nunmehr durch die Position des Knotens vorgegeben. Ist k_index negativ, so wird auf die nächste definierte Oberfläche (NURBS_SURF) verwiesen.

bound_x Beschränkungsfaktor in x-Richtung. Ist der Faktor rückgesetzt, so ist die Bewegung des Masseknotens in x-Richtung gesperrt.

bound_y Beschränkungsfaktor in y-Richtung. Ist der Faktor rückgesetzt, so ist die Bewegung des Masseknotens in y-Richtung gesperrt.

bound_z Beschränkungsfaktor in z-Richtung. Ist der Faktor rückgesetzt, so ist die Bewegung des Masseknotens in z-Richtung gesperrt.

bound_all Allgemeiner Beschränkungsfaktor. Ist der Faktor rückgesetzt, so ist die Bewegung des Masseknotens insgesamt gesperrt.

knot_coord Position des freien Masseknotens in kartesischen Koordinaten

knot_type Typ des freien Masseknotens
⇒ siehe Tabelle A.2

knot_mass Masse des freien Masseknotens

father_knot Index des Vaterknotens (übergeordneter Knoten, auf den sich der freie Knoten bezieht und dessen Bewegungsvorgaben übernommen werden)
0: kein Vaterknoten
> 0: Vaterknoten ist Nummer <father_knot> in der Liste der freien Knoten

```
ELADYN_SPRINGS    ::=
  <n_springs>     ::= unsigned
  <z_springs>     ::= unsigned
  for (all n_springs-z_springs) {        /* Basis-Verbindungselemente */
    <knot_ind1>   ::= unsigned
    <knot_ind2>   ::= unsigned
  }
  for (all z_springs) {                  /* Erweiterte Verbindungsel. */
    <knot_ind1>   ::= unsigned
    <knot_ind2>   ::= unsigned
    <sconst_z>    ::= float
    <s_type>      ::= unsigned
    <red_fact>    ::= float
    <s_param>     ::= float
  }
END ELADYN_SPRINGS
```

Spezifikation der ELADYN_SPRINGS - Struktur

n_springs Gesamtanzahl der Verbindungselemente

z_springs Anzahl der erweiterten Verbindungselemente (explizite Definition)
basic_springs = n_springs - z_springs : Anzahl der Basis-Verbindungselemente (vordefinierte Federelemente)

knot_ind1 Index des ersten Masseknotens, bezogen auf die Knoten-Liste (ELADYN_KNOT)

knot_ind2 Index des zweiten Masseknotens, bezogen auf die Knoten-Liste

sconst_z Proportionalitätsparameter des erweiterten Verbindungselements
⇒ siehe Tabelle A.3

s_type Charakteristik und Typ des Verbindungselements
⇒ siehe Tabelle A.3

red_fact Ruhelagenreduktionsfaktor ($\in$ [0,1]). Reduziert die Ruhelagenlänge des Verbindungselementes um diesen Faktor und verursacht somit eine gewisse Vorspannung

s_param Zweitparameter des Verbindungselements
⇒ siehe Tabelle A.3

B.3 Syntax des ELADYN_POLY - Primitives

Das ELADYN_POLY-Primitiv ist bezüglich der elastodynamischen Strukturen gleich aufgebaut wie das ELADYN_NURBS-Primitiv. Während bei dem letzteren jedoch eine wahlweise Darstellung der Geometrien sowohl als NURBS als auch in polygonaler Form möglich ist, beschränkt sich das ELADYN_POLY-Primitiv auf eine reine polygonale Visualisierung. Der Vorteil dieses Primitivs liegt in dem reduzierten Aufwand der Datenverwaltung und dem damit verbundenen Geschwindigkeitsvorteil sowie in einer kompakteren Primitivdefiniton.

```
ELADYN_POLY ::=
  <id>             ::= "91"
  <eladyn_type>    ::= ("1" | "2" | "3" | "4" )
  <prop_def>       ::= ELADYN_PROP
  <knot_def>       ::= ELADYN_POLYKNOT
  <spring_def>     ::= ELADYN_SPRING
  <facet_def>      ::= ELADYN_FACET
END ELADYN_POLY
```

Spezifikation des ELADYN_POLY-Primitives

Die Blöcke 'ELADYN_PROP' und 'ELADYN_SPRING' entsprechen den Blöcken des 'ELADYN_POLY'-Primitives (Anhang B.2.2). Der Block 'ELADYN_FACET' wird nur bei expliziter Definition der Objektfacetten (object_type = 5) verwendet.

```
ELADYN_POLYKNOT ::=
  <n_knots>              ::= unsigned
  <geo_knots>            ::= unsigned
  for(all n_knots) {                        /* Alle Knoten */
    <knot_coord>         ::= <x>, <y>, <z>
    if (eladyn_type == 2,4) {
      <bound_x>          ::= "0" | "1"
      <bound_y>          ::= "0" | "1"
      <bound_z>          ::= "0" | "1"
      <bound_all>        ::= "0" | "1"
    }
    if (object_type == 5)  {        /* Explizite Definition der Nachbarn */
      <tex_par_s>        ::= float
      <tex_par_t>        ::= float
      <neigh_cnt>        ::= unsigned
      for (all neigh_cnt)
        <neigh_num>      ::= unsigned
    }
    if (free_knot)  {               /* Nur bei freien Knoten */
      <knot_type>        ::= unsigned
      <knot_mass>        ::= float
      <father_knot>      ::= unsigned
    }
  }
END ELADYN_POLYKNOT

<x>, <y>, <z>            ::= float
```

Spezifikation der ELADYN_POLYKNOT - Struktur

n_knots Gesamtanzahl aller Masseknoten des elastodynamischen Netzes

geo_knots Anzahl der Masseknoten mit direktem Geometriebezug (G-Knoten, geo_knots <= n_knots). Die Differenz fz_knots = n_knots - geo_knots ist die Anzahl der Masseknoten ohne direkten Geometriebezug (F-Knoten + Z-Knoten).

knot_coord Position des freien Masseknotens in kartesischen Koordinaten

bound_x Beschränkungsfaktor in x-Richtung. Ist der Faktor rückgesetzt, so ist die Bewegung des Masseknotens in x-Richtung gesperrt.

bound_y Beschränkungsfaktor in y-Richtung. Ist der Faktor rückgesetzt, so ist die Bewegung des Masseknotens in y-Richtung gesperrt.

bound_z Beschränkungsfaktor in z-Richtung. Ist der Faktor rückgesetzt, so ist die Bewegung des Masseknotens in z-Richtung gesperrt.

bound_all Allgemeiner Beschränkungsfaktor. Ist der Faktor rückgesetzt, so ist die Bewegung des Masseknotens insgesamt gesperrt.

tex_par_s Texturparameter des Knotens in s-Richtung
Nur bei expliziter Objektdefinition müssen explizite Texturparameter angegeben werden, bei vordefinierten Objekten (object_type = 1, 2, 3) erfolgt die Erzeugung der Knoten-Texturparameter automatisch nach spezifischen Vorgaben

tex_par_t Texturparameter des Knotens in t-Richtung

neigh_cnt Anzahl der direkten Nachbarn des Knotens
Nur bei expliziter Objektdefinition muß eine Nachbarverkettung angegeben werden, bei vordefinierten Objekten (object_type = 1, 2, 3) erfolgt die Erstellung der Knoten-Nachbarschaftsbeziehungen automatisch nach spezifischen Vorgaben

neigh_num Index eines direkten Nachbarn - Verweis auf Knoten-Liste (ELADYN_POLYKNOT)

knot_type Typ des freien Masseknotens
⇒ siehe Tabelle A.2

knot_mass Masse des freien Masseknotens

father_knot Index des Vaterknotens (übergeordneter Knoten, auf den sich der freie Knoten bezieht und dessen Bewegungsvorgaben übernommen werden)
0: kein Vaterknoten
> 0: Vaterknoten ist Nummer <father_knot> in der Liste der freien Knoten

```
ELADYN_FACET ::=
  if (object_type == 5)  {          /* Explizite Definition der Facetten */
    <facet_cnt>                ::= unsigned
    for (all facet_cnt)  {
      <facet_type>             ::= unsigned
      <facet_knot1>            ::= unsigned
      <facet_knot2>            ::= unsigned
      <facet_knot3>            ::= unsigned
    }
  }
END ELADYN_FACET
```

Spezifikation der ELADYN_FACET - Struktur

facet_cnt Gesamtanzahl der Facetten (ausschließlich Dreiecke) des Objektes. Nur bei expliziter Objektdefinition muß eine Facettenliste angegeben werden, bei vordefinierten Objekten (object_type = 1, 2, 3) erfolgt die Erstellung der Facettenliste automatisch nach spezifischen Vorgaben

facet_type Spezifiziert Typ der Facette - bei der Definition noch irrelevant, wird bei Objektmodifikationen verwendet

facet_knot1 Index des ersten Masseknotens, bezogen auf die Knoten-Liste (ELADYN_POLYKNOT)

facet_knot2 Index des zweiten Masseknotens, bezogen auf die Knoten-Liste

facet_knot3 Index des dritten Masseknotens, bezogen auf die Knoten-Liste

B.4 Syntax der ELADYN_HYPER - Datei

Die elastodynamischen Eigenschaften der einzelnen Objekte werden in den jeweiligen Datensätzen festgelegt. Zusätzlich sind in einem globalen Simulationsszenario jedoch noch folgende Komponenten festzulegen:

- Verknüpfung der dynamischen Objekte untereinander
- Definition von starren Körpern, die nach außen ein dynamisches Verhalten besitzen (globaldynamische Objekte)
- Beziehung der deformierbaren Objekte und der globaldynamischen Objekte untereinander und bezüglich der (festen) Umgebung

Diese Definitionen werden in einer dem Simulationsszenario angepaßten Datei mit frei wählbarem Namen durchgeführt, die an dieser Stelle als 'ELADYN_HYPER'-Datei bezeichnet wird. Der Dateiname wird KISMET durch ein Skript-Kommando bekanntgemacht (Anhang C.1).

```
<hyper_file> = ELADYN_HYPER ::=
  <comment>        ::= string
  <hyp_type>       ::= unsigned
  <hyp_fixed>      ::= unsigned
  <hyp_conn>       ::= unsigned
  <hyp_sets>       ::= unsigned
  if (hyp_type & 0x10)        /* Globaldynamische Objekte spezifiziert */
    <hyp_gldyn>    ::= GLDYN_OBJECTS
  <hyp_fix>        ::= FIXED_KNOTS
  <hyp_connect>    ::= CONNECT_KNOTS
END ELADYN_HYPER

GLDYN_OBJECTS      ::=
  <gldyn_cnt>            ::= unsigned
  for (all gldyn_cnt)  {            /* Für alle gl.dyn. Objekte */
    <gldyn_name>         ::= string
    <gldyn_knots>        ::= unsigned
    <obj_mass>           ::= float
    <obj_damp>           ::= float
    <s_object>           ::= float
    <s_torque>           ::= float
    <inert_x>            ::= float
    <inert_y>            ::= float
    <inert_z>            ::= float
    for (all knot_cnt)         /* Für alle def. Knoten */
      coord              ::= <x>, <y>, <z>
   }
END GLDYN_OBJECTS

FIXED_KNOTS        ::=
  for (all hyp_fixed)  {            /* Für alle Geo. mit fixierten Knoten */
    <fixed_name>         ::= string
    <fix_knotcnt>        ::= unsigned
    for (all fix_knotcnt)           /* Für alle fest gebundenen Knoten */
      <fixed_num>        ::= unsigned
  }
END FIXED_KNOTS

CONNECT_KNOTS      ::=
  for (all hyp_sets)  {             /* Für alle verknüpften Objekte */
    <master_name>        ::= string
    <slave_name>         ::= string
    <knots_per_set>      ::= unsigned
    <force_fact>         ::= float
    for (all knots_per_set)  {      /* Für alle Knoten im Objektpaarsatz */
      <knot_num1>        ::= unsigned
      <knot_num2>        ::= unsigned
      if (hyp_type & 0x20) {        /* Verbindungselemente spezifiziert */
        <hyel_type>      ::= unsigned
        <hyel_const>     ::= float
        <hyel_param>     ::= float
      }
    }
  }
END CONNECT_KNOTS

  <x>, <y>, <z>          ::= float
```

Syntax der ELADYN_HYPER-Datei Spezifikation in KISMET

comment Kommentar, textuelle Charakterisierung der Datei (wird als String gelesen, muß mindestens ein Zeichen und darf keine Leerzeichen enthalten)

hyp_type Typ der ELADYN_HYPER-Datei, repräsentiert verschiedene Versionen
+ 0x10: Globaldynamische Objekte definierbar
+ 0x20: Globale Verbindungselemente definierbar

hyp_fixed Anzahl der elastodynamischen Objekte, die fest fixierte Knoten besitzen

hyp_conn Gesamtzahl der global verknüpften Knoten/Knotenpaare

hyp_sets Zahl der verknüpften Objektpaare/Sätze von Objektpaaren

gldyn_cnt Anzahl der globaldynamischen Objekte

gldyn_name Name des KISMET-Frames, welches zur Definition eines globaldynamischen Objektes herangezogen wird. Alle bezüglich dieses Frames definierten Geometrien werden als ein globaldynamisches Objekt aufgefaßt. Weitere Informationen bezüglich Modellframes sind in [KIS93] und [Kue91] aufgeführt.

gldyn_knots Anzahl der definierten festen Knoten bezüglich des globaldynamischen Objekts. Von diesen Knoten aus können Verknüpfungen zu anderen globalen Knoten durchgeführt werden.

obj_mass Gesamtmasse des globaldynamischen Objekts [kg]

obj_damp Viskose Dämpfungskonstante des globaldynamischen Objekts [N·s/m]

s_object Federkonstante der Ruhelagenfederelemente (Formerhaltung)
Zwei Federcharakteristiken sind möglich:
sconst > 0 : Federkraft proportional zur Federauslenkung [N/m]
sconst < 0 : Federkraft prop. zur quadratischen Federauslenkung [N/m²]

s_torque Federkonstante der Torsionsfederelementes am Bezugsframe des globaldynamischen Objektes [N/rad] (bezüglich der globalen Orientierung des Frames)

inert_x Rotations-Trägheitsparameter in Richtung der x-Hauptachse [kg·m²]

inert_y Rotations-Trägheitsparameter in Richtung der y-Hauptachse [kg·m²]

inert_z Rotations-Trägheitsparameter in Richtung der z-Hauptachse [kg·m²]

coord Position eines Knotens in kartesischen Koordinaten

fixed_name Name des elastodynamischen Objekts ('*.mpo'), aus dessen Knotenmenge Fixierungen vorgenommen werden

fix_knotcnt Anzahl der fixierten Knoten des ausgewählten Objekts

fixed_num Index des fixierten Knotens in der Knotenliste des Objektes

master_name Name des Objekts, welches bei der globalen Verknüpfung als Master fungiert (Positionsbestimmendes Objekt). Dies kann entweder ein elastodynamisches Objekt ('*.mpo'-Name) oder auch ein globaldynamisches Objekt (Framename) sein.

slave_name Name des Objekts, welches bei der globalen Verknüpfung als Slave fungiert (Kraftrückgebendes Objekt). Dies kann entweder eine elastodynamisches Objekt ('*.mpo'-Name) oder auch ein globaldynamisches Objekt (Framename) sein. Bei letzterem ist jedoch nur eine Verknüpfung mittels Verbindungselementen möglich.

knots_per_set Anzahl der Verknüpfungen in diesem Objektsatz

force_fact Proportionalitätsfaktor bei kraftrückgebenden Verknüpfungen ($\in [0,1]$)

knot_num1 Index des Master-Knotens in der Knotenliste des Master-Objektes

knot_num2 Index des Slave-Knotens in der Knotenliste des Slave-Objektes
Falls negativ, wird die Ruhelage des Master-Knotens als Slave-Knotenposition angenommen (nur bei Verknüpfung mit Verbindungselementen sinnvoll)

hyel_type Charakteristik und Typ des Verbindungselements
$\Rightarrow$ siehe Tabelle A.3
Falls Null, wird zwischen beiden Knoten kein Verbindungselement eingefügt, sondern es erfolgt eine direkte Kopplung. Hierbei wirkt der Masterknoten positionsvorgebend, der Slaveknoten gibt eine resultierende Kraft an den Masterknoten zurück. Genauso wird auch bei unspezifizierten Verbindungselementen verfahren.

hyel_const Proportionalitätsparameter des Verbindungselements
$\Rightarrow$ siehe Tabelle A.3

hyel_param Zweitparameter des Verbindungselements
$\Rightarrow$ siehe Tabelle A.3

Anhang C: Spezifikation der KISMET Skript-Kommandos

Skript-Kommandos sind Befehlszeilen in einer textuellen Datei, über die interaktiv Einfluß auf die Modelle und die Simulation ausgeübt werden kann. Skripte können sowohl über die Bedienoberfläche als auch durch Tastendruck, Fußschalterbetätigung oder durch Eintreten eines bestimmten Ereignisses aufgerufen werden.

In KISMET existiert eine Vielzahl an Skript-Kommandos mit verschiedensten Funktionen (beispielsweise zur Modellmodifikation, Steuerung des Simulationsablaufs, Roboterdefinitionen). An dieser Stelle sollen nur diejenigen Skript-Befehle aufgeführt werden, die für die Simulation deformierbarer Objekte unmittelbare Relevanz besitzen. Ansonsten wird auf den KISMET-Leitfaden [KIS93] sowie [Kue91] verwiesen.

Notation:

< ... >	variable Parameterelemente
[.. | ..]	Trennung von Parameteralternativen
{ ... }	optionale Angabe, Elemente zwischen { } dürfen entfallen

C.1 Befehle zur Definition deformierbarer Objekte

Diese Skript-Befehle dienen zur Modifikation der elastodynamischen und geometrischen Daten deformierbarer Objekte sowie deren Visualisierung, der globalen Verknüpfung und der Anpassung von Interaktionsparametern.

eladyn_geo_def [<geo_name> | ALL] <flags> {<value1> <value2> <value3> <value4>}

Das Skript-Kommando definiert die gesamten Objekt-Attribute für ein oder alle elastodynamische Objekte.

<geo_name> [*string*]
Definiert den Dateinamen des elastodynamischen Objektes im KISMET-spezifischen Format ('*.mpo'). Durch Angabe von 'ALL' werden alle deformierbaren Objekte erfaßt

<flags> [*unsigned*]
Attributfeld, definiert objektspezifisches Verhalten, Berechnung und Darstellung
⇒ Tabelle A.1

<value1 .. 4> [*float*]
Simulationsspezifische Modifikation von Parametern (i.A. irrelevant)

eladyn_geo_def [<geo_name> | ALL] <token> [on | off]

Das Skript-Kommando definiert ein einzelnes Objekt-Attribut für ein oder alle elastodynamischen Objekte. Durch Angabe von 'on' wird das spezifizierte Attribut aktiviert, 'off' deaktiviert es.

<geo_name> [*string*]
Definiert den Dateinamen des elastodynamischen Objektes im KISMET-spezifischen Format ('*.mpo'). Durch Angabe von 'ALL' werden alle deformierbaren Objekte erfaßt.

<token> [*string*]
Schlüsselwort zur Definition des Objekt-Attributs im Attributfeld
⇒ Tabelle A.1

eladyn_hyper_name <hyper_file>

Spezifiziert den Namen der ELADYN_HYPER-Datei, in welcher die objektübergreifenden Verknüpfungen definiert sind (Anhang B.4).

<hyper_file> [*string*]
Definiert den Dateinamen der ELADYN_HYPER-Datei (lokalisiert im KISMET-Modellverzeichnis '$kis_home/envlib')

eladyn_show_def [0 | 1]

Skript-Befehl für die Darstellung der Elemente des elastodynamischen Netzes (0-Darstellung aus, 1-Darstellung ein). Die Masseknoten werden als Tetraeder repräsentiert, die in Abhängigkeit ihrer Attribute und ihres Interaktionszustandes verschieden gefärbt sind. Die Verbindungselemente werden durch Linien zwischen den entsprechenden Knoten dargestellt. Außerdem werden die Kraftvektoren der Knoten und die Orientierung der Zentralknoten visualisiert.

eladyn_model_reset

Setzt die Simulationsstruktur mit allen elastodynamischen und globaldynamischen Objekten auf den Ausgangszustand zurück. Eventuell interaktiv geänderte Modellparameter bleiben jedoch erhalten. Modellmanipulationen werden revidiert, alle Interaktionsstrukturen gelöscht, die gesetzten Klemmen entfernt.

eladyn_plaus_param <token> <flags> <val>

Mit diesem Skript-Befehl können diverse Parameter der Plausibilitätsprüfung modifiziert werden, um die Interaktionsauswertung an spezifische Erfordernisse des Simulationsszenarios anzupassen.

<token> [*string*]
Schlüsselwort zur Definition der Stufe und Lokalität der Plausibilitätsprüfung
⇒ Tabelle A.5

<flags> [*unsigned*]
Definiert Kriterien bei der Plausibilitätsprüfung, Typ der Auswertung
⇒ Tabelle A.4

<val> [*float*]
Schwellwert bei Auswertung der spezifizierten Stufe der Plausibilitätsprüfung

C.2 Befehle zur Simulationsunterstützung

Diese Skript-Kommandos dienen im wesentlichen zur Simulationsunterstützung von KISMET als interaktive Trainingsumgebung. Hierzu gehören die Definition verschiedener 'Roboter' als Instrumente für die Minimal-Invasive Chirurgie mit zusätzlichen geometrischen Spezifikationen sowie die Festlegung der Kommunikationsparameter für die Positionsvorgabe durch die Phantombox.

eladyn_calc [on | off]

Schaltet die elastodynamischen Berechnungen sowie die Interaktionsbehandlung ein (beziehungsweise aus). Wichtig ist die Deaktivierung insbesondere bei nachfolgenden Modellmodifikationen, um inkonsistente Zustände der Modelldaten zu verhindern. Im Mehrprozessorbetrieb kann dies ansonsten zu Systemabstürzen führen.

rob_ela_interact <int_jnt> <test_funct> <testval1> <testval2> <inst_type>

Bestimmt das Verhalten des aktuellen Roboters (Skript-Befehl 'ROBACT') als MIC-Instrument. Insbesondere die Effektorwirkung wird spezifiziert.

<int_jnt> [*unsigned*]
Definiert den Effektorfreiheitsgrad, der die spezifische Wirkung des Instruments auslöst (Index des Freiheitsgrades in der Roboterdefinition)

<test_funct> [*unsigned*]
Spezifiziert die Testfunktion, die die Effektorwirkung auslöst. Der aktuelle Wert des Effektorfreiheitsgrades <int_jnt> ist hierbei <jnt_val>, <testval1> und <testval2> sind benutzerdefinierte Schwellwerte der Testfunktion:
1: jnt_val ≥ testval1
2: jnt_val < testval1
3: testval1 ≤ jnt_val ≤ testval2
4: (jnt_val < testval1) oder (jnt_val > testval2)

<inst_type> [*unsigned*]
Definiert den Typ des simulierten MIC-Instrumentes und damit die Art der Effektorwirkung. Folgende Instrumententypen sind spezifiziert:
0: Greifer
1: Clip-Applikator
2: Schere
3: Koagulations-Haken
4: Koagulations-Zange

ela_clip_def <clip_file> <trans_rx> <trans_ry> <trans_rz> <trans_tx> <trans_ty> <trans_tz>

Mit einem simulierten MIC-Instrument, dem Clip-Applikator, können an deformierbaren Objektstrukturen Klemmen gesetzt werden. Die Geometrie und Lage der Klemmen wird über dieses Skript-Kommando spezifiziert.

<clip_file> [*string*]
Definiert die Geometrie-Datei der Klemme im KISMET-spezifischen Format ('*.mpo')

<trans_rx> [*float*]
Transformationsparameter der Geometrie - Rotation um x-Achse

<trans_ry> [*float*]
Transformationsparameter der Geometrie - Rotation um y-Achse

<trans_rz> [*float*]
Transformationsparameter der Geometrie - Rotation um z-Achse

<trans_tx> [*float*]
Transformationsparameter der Geometrie - Translation in x-Richtung

<trans_ty> [*float*]
Transformationsparameter der Geometrie - Translation in y-Richtung

<trans_tz> [*float*]
Transformationsparameter der Geometrie - Translation in z-Richtung

rob_ela_geom <geom1> <geom2> <geom3> <geom4>

Bei der starken Abstrahierung der Interaktionserkennung bleibt die definierte Geometrie der Instrumente weitgehend unberücksichtigt. Für realistische, aber dennoch echtzeitfähige Interaktionen werden dem aktuellen MIC-Instrument (Skript-Befehl 'ROBACT') grobe geometrische Attribute zugewiesen.

<geom1> [*float*]
Definiert den Schaftdurchmesser des simulierten Instrumentes [mm]

<geom2> [*float*]
Durchschnittlicher Abstand der äußeren Effektoroberfläche von der virtuellen Effektorachse ([mm], in Effektorbewegungsrichtung)

rob_ela_shm <send_type> <re_shmkey> <re_buflen>

Über Skript-Befehle werden auch die Kenndaten der Kommunikation mit anderen Prozessen festgelegt. Dieses Kommando definiert einen gemeinsamen Speicherbereich (Shared-Memory), über den die am aktuellen Instrument angreifenden Kräfte exportiert werden. Alle Kräfte werden hierbei so umgerechnet, als ob sie auf die Effektorspitze wirken. Die Ausgabe ist insbesondere für eine eventuelle Kraftrückkopplung mittels eines kraftreflektierenden Eingabegerätes notwendig.

<send_type> [*unsigned*]
Definiert die Art des Datenaustauschs (momentan irrelevant)

<re_shmkey> [*unsigned*]
Schlüssel zur eindeutigen Identifizierung des gemeinsamen Speicherbereichs, wird von beiden kommunizierenden Prozessen benötigt

<send_type> [*unsigned*]
Größe des gemeinsamen Speicherbereichs (in Byte)

bin_ela_shm <send_type> <be_shmkey> <be_buflen>

Dieses Kommando definiert einen gemeinsamen Speicherbereich (Shared-Memory), über den binäre Informationen in Form von externen Eingabe- und Bedienmedien importiert werden. Für die MIC-Trainingsumgebung ist dies beispielsweise in Form von Fußschaltern möglich, deren Aktivierung dann verschiedene (allgemeine und auch simulierte chirurgische) Funktionen auslöst.

<send_type> [*unsigned*]
Definiert die Art des Datenaustauschs (momentan irrelevant)

<re_shmkey> [*unsigned*]
Schlüssel zur eindeutigen Identifizierung des gemeinsamen Speicherbereichs, wird von beiden kommunizierenden Prozessen benötigt

<send_type> [*unsigned*]
Größe des gemeinsamen Speicherbereichs (in Byte)

hot_message <text>

Anzeige von System- und Simulationsmeldungen auf dem Bildschirm (für etwa 5 Sekunden). Bedienvorgänge können bestätigt und Trainingsergebnisse angezeigt werden.

<text> [*string[..]*] Anzuzeigender Text

Verzeichnis der Abbildungen

Literaturverzeichnis

[Bar84] A. Barr
Global and Local Deformations of Solid Primitives
in: Computer Graphics, Proc. SIGGRAPH'84, ACM, S. 21-30, 1984

[BBB87] R. Bartels, J. Beatty, B. Barsky
An Introduction to Splines for Use in Computer Graphics & Geometric Modeling
Morgan Kaufmann Publishers, Los Altos, 1987

[BGT96] R. Baumann, D. Glauser, D. Tappy, C. Baur, R. Clavel
Force Feedback for Virtual Reality Based Minimally Invasive Surgery Simulator
in: Health Care in the Information Age, MMVR'96, San Diego, S. 564-579, 1996

[BP91] W. Boehm, H. Prautzsch
Numerical Methods
Vieweg, Braunschweig, 1991

[BS87] I. Bronstein, K. Semendjajew
Taschenbuch der Mathematik
Verlag Harri Deutsch, 1987

[BT75] P. Bui-Tuong
Illumination for Computer-Generated Pictures
Communications of the ACM, 18(6), S. 311-317, 1975

[Bue95] G. Bueß
Operationslehre der endoskopischen Chirurgie
Springer-Verlag, Berlin, 1995

[BW76] K.-J. Bathe, E.L. Wilson
Numerical methods in finite elemente analysis
Prentice-Hall, New Jersey, 1976

[CDC96] S. Cotin, H. Delingette, J.M. Clément, V. Tassetti, J. Marescaux, N. Ayache
Volumetric Deformable Models for Simulation of Laparoscopic Surgery
in: Proc. Computer Assisted Radiology CAR'96, Paris, Elsevier, S. 793-798, 1996

[CZ92] D. Chen, D. Zeltzer
Pump it up: Computer Animation of a Biomechanically Based Model of Muscle Using the Finite Element Method
in: Computer Graphics, Proc. SIGGRAPH'92, ACM, S. 89-98, 1992

[Dai88] F. Dai
Collision-Free Motion of an Articulated Kinematic Chain in a Dynamic Environment
IEEE Computer Graphics & Applications, S. 70-74, 1988 .

[DCH88] R. Drebin, L. Carpenter, P. Hanrahan
Volume Rendering
in: Computer Graphics, Proc. SIGGRAPH'88, ACM, S. 65-74, 1988

[Deu96] O. Deussen
Untersuchung effizienter Verfahren zur Bewegungssimulation deformierbarer Körper
Dissertation, Universität Karlsruhe, 1996

[Die89] R. Diehl
Approximationsmodelle und ihr Einsatz bei der Kollisionsprüfung bewegter Volumenmodelle
Fortschrittsberichte VDI Reihe 10, Nr. 108, 1989

[DK95] O. Deussen, Ch. Kuhn
Echtzeitsimulation deformierbarer Objekte über nodale Modelle
in: Proc. 'Integration von Bild, Modell und Text`95', ASIM-Mitteilungen Nr.46, Magdeburg, S. 117-128, 1995

[DKT95] O. Deussen, L. Koppelt, P. Tücke
Using simulated annealing to obtain good approximations of deformable bodies
in: Computer Animation and Simulation '95, Springer-Verlag, S. 30-43, 1995

[Dum95] A. Dumay
Triage Simulation in a Virtual Environment
in: Interactive Technology & The new Paradigm for Healthcare, Proc. MMVR'95, San Diego, S. 101-111, 1995

[Far88] G. Farin
Curves and Surfaces for Computer Aided Geometric Design
Academic Press, Boston, 1988

[Far92] G. Farin
From Conics to NURBS: A Tutorial and Survey
IEEE Computer Graphics & Applications, S. 78-86, 1992

[FDF90] J.D. Foley, A. van Dam, S.K. Feiner, Hughes
Computer Graphics - Principles and Practice
Addison-Wesley Publishing, 1990

[Foe90a] O. Föllinger
Regelungstechnik
Hüthig-Verlag, Heidelberg, 1990

[Foe90b] O. Föllinger
Lineare Abtastsysteme
Oldenbourg Verlag, München, 1990

[Fun90] Y.C. Fung
Biomechanics
Springer-Verlag, New York, 1990

[FvD84] J.D. Foley, A. van Dam
Fundamentals of Interactive Computer Graphics
Addison-Wesley Publishing, 1984

[GAS94] A. García-Alonso, N. Serrano, J. Flaquer
Solving the Collision Detection Problem
in: IEEE Computer Graphics & Applications, S. 36-43, 1994

[GMM96] G. Glombitza, M.H. Makabe, H.P. Meinzer
Formorientierte Korrektur von regionenorientierten Bildanalysemethoden
in: Proc. Workshop 'Digitale Bildverarbeitung in der Medizin', Universität Freiburg, S. 120-125, 1996

[Gou71] H. Gouraud
Continuous Shading of Curved Surfaces
IEEE Transactions on Computers, 20 (6), S. 623-629, 1971

[GPS93] F. Götz, A. Pier, E. Schippers, V. Schumpelick
Color Atlas of Laparoscopic Surgery
Thieme Verlag, Stuttgart, 1993

[Gra87] H. Grabowski
Algorithmische Kollisionsprüfung für die Simulation von Bewegungsabläufen
DFG-Abschlußbericht 02/87, Universität Karlsruhe, 1987

[GTT89] J.P. Gourret, N. Thalmann, D. Thalmann
Simulation of Object and Human Skin Deformations in a Grasping Task
in: Computer Graphics, Proc. SIGGRAPH '89, ACM, S. 21-30, 1989

[Hah88] J. Hahn
Realistic Animation of Rigid Bodies
in: Computer Graphics, Proc. SIGGRAPH '88, ACM, S. 299-308, 1988

[Hec86] P. Heckbert
Survey of Texture Mapping
IEEE Computer Graphics & Applications, S. 56-67, 1986

[HHK92] W. Hsu, J. Hughes, H. Kaufman
Direct Manipulation of Free-Form Deformations
in: Computer Graphics, Proc. SIGGRAPH '92, ACM, S. 177-184, 1992

[HK96] M. Hübner, U. Kühnapfel
Real-time volume visualization of medical image data for diagnostic and navigational purposes in computer aided surgery
in: Proc. Computer Assisted Radiology CAR'96, Paris, Elsevier, S.751-756, 1996

[HL89] J. Hoschek, D. Lasser
Grundzüge der geometrischen Datenverarbeitung
Teubner-Verlag, Stuttgart, 1989

[Hoe87] K.H. Höhne
3D-Bildverarbeitung und Computergraphik in der Medizin
Informatik-Spektrum Vol. 10, S. 192-204, 1987

[Hon96] Hon, David
Medical Reality and Virtual Reality
in: Health Care in the Information Age, MMVR'96, San Diego, S. 327-341, 1996

[HS85] E. Hundt, G. Schwierz
Verfahren und Systeme der Computertomographie
Informatik-Spektrum Vol. 8, S. 273-282, 1985

[Hue71] Hütte
Physikhütte, Band I: Mechanik
Verlag Wilhelm Ernst, Berlin, 1971

[JBW96] R. Johnston, S. Bhoyrul, L. Way, R. Satava, K. McGovern, J. Fletcher u.a.
Assessing a Virtual Reality Surgical Skills Simulator
in: Health Care in the Information Age, MMVR'96, San Diego, S. 608-617, 1996

[JT80] C. Jackins, S. Tanimoto
Octrees and their Use in Representing three-dimensional objects
Computer Graphics and Image Processing, 14 (3), S. 249-270, 1980

[KGE75] R. Kenedi, T. Gibson, J. Evans, J. Barbenel
Tissue Mechanics
Phys.Med.Biol., 20(5), S. 699-717, 1975

[KHK75] J. Köhler, R. Höwelmann, H. Krämer
Analytische Geometrie und Abbildungsgeometrie in vektorieller Darstellung
Diesterweg-Salle, Frankfurt, 1975

[KIS93] Produktdokumentation
KISMET: User-Data Specification
Bericht des Forschungszentrum Karlsruhe, IRE/6/010/92, 1993

[KKD95] Ch. Kuhn, U. Kühnapfel, O. Deussen
Echtzeitsimulation deformierbarer Objekte zur Ausbildungsunterstützung in der Minimal-Invasiven Chirurgie
in: Proc. Int. Workshop MVD`95, Bad Honnef, infix-Verlag, S. 169-178, 1995

[KKH96] U. Kühnapfel, Ch. Kuhn, M.Hübner, H.-G. Krumm
VR Technology based Minimally Invasive Surgery Training using the KISMET Software
in: Proc. IMAGINA '96, Monte Carlo, 1996

[KKN95] U. Kühnapfel, Ch. Kuhn, B. Neisius
Der 'Karlsruher Endoskopietrainer', ein auf 'Virtual Reality' Techniken basierendes MIC-Trainingssystem
Bericht FZKA-5670, Karlsruhe, S. 187, 1995

[KKK94] U. Kühnapfel, Ch. Kuhn, H.-G. Krumm, M.Hübner
CAD-based Simulation and Modelling for Endoscopic Surgery
in: Proc. MedTech, SMIT`94, Berlin, 1994

[KKK95] U. Kühnapfel, H.-G. Krumm, Ch. Kuhn, M.Hübner, B. Neisius
Endosurgery Simulations with KISMET: A flexible tool for Surgical Instrument Design, Operation Room Planning and VR Technology based Abdominal Surgery Training
in: Proc. VR World`95, Stuttgart, Computerwoche-Verlag, S. 165-171, 1995

[KKK96] Ch. Kuhn, U. Kühnapfel, H.-G. Krumm, B. Neisius
A 'Virtual Reality'-based Training System for Minimally Invasive Surgery
in: Proc. Computer Assisted Radiology CAR'96, Paris, Elsevier, S. 764-769, 1996

[Kli94] A. Klingert
Rekonstruktion von geometrischen 3D-Objekten durch Interpolation von Konturdaten
Dissertation, Universität Karlsruhe, Shaker-Verlag, 1994

[KN93] U. Kühnapfel, B. Neisius
CAD-Based Graphical Computer Simulation in Endoscopic Surgery
in: Endoscopic Surgery and Allied Technologies, Georg Thieme Verlag, Vol. 1, No.3, S. 181-184, 1993

[KN94] U. Kühnapfel, B. Neisius
Realtime Graphical Computer Simulation for Endoscopic Surgery
in: Proc. Medicine meets Virtual Reality II, San Diego, IOS Press, S. 114-116, 1994

[KSH95] R. Klein, R. Sonntag, T. Hüttner
Datenreduktion von Radiositynetzen zum Einsatz globaler Beleuchtungen in Anwendungen der virtuellen Realität
in: Proc. Int. Workshop MVD`95, Bad Honnef, infix-Verlag, S. 99-106, 1995

[Kue91] U. Kühnapfel
Grafische Realzeitunterstützung für Fernhandhabungsvorgänge in komplexen Arbeitsumgebungen im Rahmen eines Systems zur Steuerung, Simulation und Off-Line-Programmierung
Dissertation, Universität Karlsruhe, 1991 (auch: Bericht KFK-5052 (1992))

[KW91] K. Knothe, H. Wessels
Finite Elemente
Springer-Verlag, Berlin, 1991

[Lar86] W. Larrabee
A Finite Element Model of Skin Deformation
Laryngoscope, S. 399-412, April 1986

[LW94] H. Lamousin, W. Waggenspack
NURBS-Based Free-Form Deformations
IEEE Computer Graphics and Applications, S. 59-65, 1994

[MGJ96] K. McGovern, R. Johnston
The Role of Computer-Based Simulation for Training Surgeons
in: Health Care in the Information Age, MMVR'96, San Diego, S. 342-345, 1996

[MP89] G. Miller, A. Pearce
Globular Dynamics: A Connected Particle System for Animating Viscous Fluids
Computer & Graphics, 13(3), S.305-309, 1989

[MRM96] D. Meglan, R. Raju, G. Merril, J. Merril, B. Nguyen u.a.
The TELEOS Virtual Environment Toolkit for Simulation-Based Surgical Education
in: Health Care in the Information Age, MMVR'96, San Diego, S. 346-351, 1996

[MS94] T. Massie, J. K. Salisbury
The PHANToM Haptic Interface: A Device for Probing Virtual Objects
Proc. ASME Winter Annual Meeting, Chicago, Nov. 1994

[MW88] M. Moore, J. Wilhelms
Collision Detection and Response for Computer Animation
in: Computer Graphics, Proc. SIGGRAPH '88, ACM, S. 289-298, 1988

[MYV93] J. Maillot, H. Yahia, A. Verroust
Interactive Texture Mapping
in: Computer Graphics, Proc. SIGGRAPH '93, ACM, S. 27-34, 1993

[NH94] B.M. Nigg, W. Herzog
Biomechanics of the Muscolo-Skeletal System
John Wiley & Sons, Chichester, 1994

[NPW96] S. Nöh, E. Pelikan, S. Wegner, T. Tolxdorff
Ein Ansatz zur objektorientierten Bildsegmentierung
in: Proc. Workshop 'Digitale Bildverarbeitung in der Medizin', Universität Freiburg, S.8-15, 1996

[PB88] J. Platt, A. Barr
Constraint Methods for Flexible Models
in: Computer Graphics, Proc. SIGGRAPH '88, ACM, S. 279-288, 1988

[Pie91] L. Piegl
On NURBS: A Survey
IEEE Computer Graphics & Applications, S. 55-71, 1991

[PS90] A. Pentland, S. Sclaroff
Closed-Form Solutions for Physically-Based Shape Modelling and Recognition
M.I.T. Technical Report No.135, März 1990

[RB96] D. Rueckert, P. Burger
Shape-based Tracking and Analysis of the Aorta in Cardiac MR Images Using Geometrically Deformable Templates
in: Proc. Computer Assisted Radiology CAR'96, Paris, Elsevier, S. 274-279, 1996

[Ree83] W. Reeves
Particle Systems - A Technique for Modeling a Class of Fuzzy Objects
Computer Graphics, 17(3), S. 359-376, 1983

[Rot82] S. D. Roth
Ray Casting for Modeling Solids
Computer Graphics and Image Processing 18, S. 109-144, 1982

[RRP96] K. Reinig, C. Rush, H. Pelster, V. Spitzer, J. Heath
Real-Time Visually and Haptically Accurate Surgical Simulation
in: Health Care in the Information Age, MMVR'96, San Diego, S. 542-545, 1996

[Rup95] M. Rupp
Konvertierung von NURBS in polygonale Modelle zum Einsatz in einer grafischen Echtzeit-Simulationsumgebung
Diplomarbeit, Fak. für Informatik, Universität Karlsruhe, 1995

[RV83] A. Requicha, H. Voelcker
Solid Modelling: Current Status and Research Directions
IEEE Computer Graphics & Applications 3, S. 25-37, 1983

[Sch88] F. Scheid
Numerical Analysis
Schaum's Outline, McGraw-Hill, 1988

[Sch96] M. Schmid
Virtual Reality in der Produktentwicklung
in: Computer Zeitung Nr.23, S. 20, 1996

[SGI91] Firmendokumentation
Graphics Library Programming Guide
Silicon Graphics Inc., Mountain View, 1991

[SM95] F. Schöffel, S. Müller
Radiosity in interaktiven virtuellen Welten
in: Proc. Int. Workshop MVD`95, Bad Honnef, infix-Verlag, S. 107-119, 1995

[SP86] T. Sederberg, S. Parry
Free-Form Deformations of Solid Geometric Models
in: Computer Graphics, Proc. SIGGRAPH '86, ACM, S. 151-160, 1986

[Spi76] M. Spiegel
Allgemeine Mechanik - Theorie und Anwendung
McGraw-Hill, New York, 1976

[SZ92] R. Szeliski, D. Tonnesen
Surface Modeling with Oriented Particle Systems
in: Computer Graphics, Proc. SIGGRAPH '92, ACM, S. 185-194, 1992

[SWF93] J. Snyder, A. Woodbury, K. Fleischer, B. Currin, A. Barr
Interval Methods for Multi-Point Collisions between Time-Dependent Curved Surfaces
in: Computer Graphics, Proc. SIGGRAPH'93, ACM, S. 321-334, 1993

[TF88] D. Terzopoulos, K. Fleischer
Deformable Models
in: The Visual Computer, Springer-Verlag, S. 125-150, 1988

[TK95] H. Tek, B. Kimia
Automatic Volumetric Segmentation of Three-Dimensional Medical Images
in: Proc. Computer Assisted Radiology '95, Berlin, Springer, S. 165-170, 1995

[TM91] D. Terzopoulos, D. Metaxas
Dynamic 3D Models with Local and Global Deformations: Deformable Superquadrics
IEEE Transactions on Pattern Analysis and Machine Intelligence, 13 (7), S. 703-714, 1991

[TPB87] D. Terzopoulos, J. Platt, A. Barr, K. Fleischer
Elastically Deformable Models
in: Computer Graphics, Proc. SIGGRAPH '87, ACM, S. 205-214, 1987

[TW88] D. Terzopoulos, A. Witkin
Deformable Models: Physically Based Models with Rigid and Deformable Components
IEEE Computer Graphics & Applications, S. 41-51, 1988

[TW90] D. Terzopoulos, K. Waters
Physically-based Facial Modelling, Analysis, and Animation
The Journal of Visualization and Computer Animation Vol.1, S. 73-80, 1990

[TW91] D. Terzopoulos, K. Waters
Techniques for Realistic Facial Modeling and Animation
in: Computer Animation '91, Springer-Verlag, Tokyo, S. 59-74, 1991

[UOT83] T. Uchiki, T. Ohashi, M. Tokoro
Collision Detection in Motion Simulation
Computer & Graphics Vol.7, No.3-4, S. 285-293, 1983

[Voe95] S. Völter
Virtual Reality in der Medizin I
GeSI - Verlag, Mannheim, 1995

[VT95] P. Volino, N. Thalmann
Collision and Self-Collision Detection: Efficient and Robust Solutions for Highly Deformable Surfaces
in: Computer Animation and Simulation '95, Maastricht, Springer, S. 55-65, 1995

[Wat92] K. Waters
A physical model of facial tissue and muscle articulation derived from computer tomography data
in: Visualization in Biomedical Computing, SPIE Vol.1808, S. 574-583, 1992

[Whi80] T. Whitted
An Improved Illumination Model for Shaded Display
Communications of the ACM, 23(6), S. 343-349, 1980

[ZDS92] C.E. Zöckler, K. Draese, J. Schopohl
Spezielle Chirurgie der Gallenwege
Karger-Verlag, Freiburg, 1992

[ZM83] O.C. Zienkiewicz, K. Morgan
Finite elements and approximations
John Wiley & Sons, New York, 1983

[ZMF95] R. Ziegler, W. Müller, G. Fischer, M. Göbel
A Virtual Reality Medical Training System
in: Proc. Computer Vision, Virtual Reality and Robotics in Medicine, CVRMed'95, Nizza, Springer-Verlag, S. 282-286, 1995