KERNFORSCHUNGSZENTRUM

KARLSRUHE

August 1976 KFK 2305

Institut für Reaktorentwicklung

System und Sprache zur Behandlung graphischer Information im rechnergestützten Entwurf

R. Schuster

GESELLSCHAFT
FÜR
KERNFORSCHUNG M.B.H.

KARLSRUHE

GESELLSCHAFT FÜR KERNFORSCHUNG M. B. H.

KARLSRUHE

KERNFORSCHUNGSZENTRUM KARLSRUHE

KFK 2305

Institut für Reaktorentwicklung

System und Sprache zur Behandlung graphischer Information im rechnergestützten Entwurf x)

Richard Schuster

Gesellschaft für Kernforschung m.b.H., Karlsruhe

x) Als Dissertation genehmigt von der Fakultät für Maschinenbau der Universität Karlsruhe (TH)

Kurzfassung:

Das Subsystem GIPSY (Graphical Information Processing System) stellt im Rahmen des Systems für das rechnergestützte Entwickeln und Konstruieren REGENT graphische Fähigkeiten für die Behandlung zwei- und dreidimensionaler Objekte bereit. Die Formulierung der graphischen Aufgaben ist in einer um graphische Anweisungen und Datentypen erweiterten, auf PL/1 basierenden Sprache möglich.

Räumliche Objekte, deren Darstellung durch analytisch bestimmte Durchdringungs- und Umrißkurven unter Beachtung der Sichtbarkeit erfolgt, können von ebenen, kugeligen, zylindrischen und kegeligen Flächen begrenzt sein.

Abstract:

System and Language for Graphical Applications in a CAD-System

The subsystem GIPSY (Graphical Information Processing System) provides graphical features for working with two- and three-dimensional objects within the CAD-System REGENT. For the formulation of graphical problems an extension of PL/1 is available, which adds graphical data types and operations.

Three dimensional objects are represented by analytically determined intersection and contour curves of plane, spherical, cylindrical and conical surfaces. Hidden line elimination is supported for all objects and projections.

Inhaltsverzeichnis Seite

1. Einleitung

Zeichnungen sind im wissenschaftlich-technischen Bereich häufig genutzte Informationsträger. Sie stellen nicht selten für Konstrukteure und Ingenieure die anschaulichste, kompakteste und einzig zweifelsfreie Grundlage der Kommunikation dar.

Die Arbeiten zur Erstellung dieser Unterlagen beanspruchen einen großen Teil des im Konstruktionsbereich anfallenden Aufwandes. Da die konstruktiven Tätigkeiten maßgeblich die Durchlaufzeit eines Produktes beeinflussen, ist der in diesem Bereich erzielbare Rationalisierungseffekt stark von der Erleichterung oder der vollständigen Automatisierung der Zeichnungserstellung abhängig.

Durch den hohen Anteil an manuell schematischen Vorgängen, den die zeichnerischen Tätigkeiten enthalten, sind sie für den Einsatz der elektronischen Datenverarbeitung (EDV) prädestiniert.

Im Vergleich zu der Verbreitung, die die EDV im kaufmännisch-organisatorischen Bereich oder auch bei der Berechnung technischer Probleme gefunden hat, steckt die Nutzung der graphischen Datenverarbeitung zur Zeichnungserstellung noch in den Anfängen.

Ein wichtiges Hindernis für die Realisierung dieser Anwendungen bedeutet das Fehlen von adäquater Techniken zur Formulierung graphischer Probleme in einer maschinenlesbaren Form. Für die Lösung mathematischer Probleme bieten die höheren Programmiersprachen angemessene Hilfsmittel an. Die Codierung eines algebraischen Ausdruckes in einer der Sprachen ALGOL, FORTRAN oder PL/1 unterscheidet sich nur unwesentlich von dem in der Mathematik verwendeten Formalismus.

Für die Entwicklung von Zeichnungserstellungs- oder gar Konstruktionssystemen stehen vergleichbare Fähigkeiten zur Programmierung der graphischen Aufgaben bisher nicht zur Verfügung. Die Möglichkeit zur Nutzung graphischer Fähigkeiten beschränkt sich auf die von den Herstellern graphischer Ausgabegeräte angebotene Software. Diese auf der Unterprogrammtechnik beruhende Methode ist für eine sich an der Vorgehensweise zeichnerischer Tätigkeiten orientierende Formulierung wenig geeignet.

Zur Lösung graphischer Aufgaben ist eine Eingabeform anzustreben, die eine Nutzung der anschaulich vertrauten Objekte als Elemente einer Modellbeschreibung möglich macht. Dieser Katalog sollte über diejenigen geometrischen Elemente der Ebene und des Raumes verfügen, in denen ein Konstrukteur "denkt", wenn er eine Zeichnung erstellt.

Auf der Basis dieser Grundelemente, wie Punkte, Polygone, Kreise oder auch Ebenen, Kugeln, Zylinder und Kegel, sind Verfahren zu entwickeln, die eine effektive und flexible Verarbeitung dieser graphischen Information unterstützen. Diese sollten es erlauben, durch geeignete Anweisungen die vertrauten graphischen oder konstruktiven Arbeitsschritte nachzuvollziehen.

Die beliebige Kombination von Gestaltungsgrundelementen erlaubt dann die Erstellung betriebsunabhängiger Programmsysteme zur Behandlung von Werkstückgeometrien mit großer Variationsbreite.

In Ermanglung derartiger Fähigkeiten sind viele der heute verfügbaren Systeme auf die Behandlung spezieller Werkstückarten mit ausgezeichneten Symmetrieeigenschaften, wie z.B. Rotationsteile, in rein zweidimensonaler Darstellung beschränkt.

Die dreidimensionale Bauteilbeschreibung ist jedoch erforderlich, wenn die automatisierte Zeichnungserstellung ein breiteres Teilespektrum erfassen soll. Die zur Verdeutlichung der Werkstückgeometrie erforderlichen Ansichten können nur aus einer rechnerinternen räumlichen Repräsentation des Objektes gewonnen werden.

Für die Erzeugung von Darstellungen eines dreidimensionalen Objektes ergeben sich neben der Formulierung der Eingabe Schwierigkeiten bei der Bestimmung der durch Schnittbildung entstehenden Raumkurven und der Anwendung von Visibilitätskriterien. Beide Probleme sind durch die bisher verwendeten Verfahren nur mit großem numerischen Aufwand lösbar.

Trotz ihrer Bedeutung darf die Zeichnungserstellung jedoch nicht isoliert betrachtet werden. Komplexe Entwurfsaufgaben erfordern zu ihrer Lösung in allen Phasen die Beschaffung, Speicherung und die Wiedergabe von Informationen verschiedener Fachgebiete. Für die graphischen Fähigkeiten ergibt sich daraus die Notwendigkeit ihrer Einbeziehung in ein umfassendes System zur rechnerunterstützten Lösung aller Teilschritte und -aufgaben des Entwurfsprozesses.

Im Sinne einer effektiven Auftragsbearbeitung sollten deshalb während des Konstruktionsprozesses anfallende Daten ohne manuellen Eingriff für die rechnerinterne Weiterverarbeitung verfügbar bleiben. Die aus der Rechneranwendung erreichbaren Einsparungen gehen sonst durch wiederholt anfallenden Eingabeaufwand wieder verloren.

Durch diese Verknüpfung ist es dann möglich, die mittels Auslegungsrechnung gewonnenen Größen direkt für die Erstellung von Fertigungsunterlagen eines Werkstückes zu nutzen.

Aus dieser Problemstellung strebt die vorliegende Arbeit ein System zur Behandlung allgemeiner graphischer Information (GIPSY = Graphical Information Processing System) mit folgenden Merkmalen an:

- Ein umfangreicher Katalog an graphischen Fähigkeiten, der die Körperbehandlung einschließt, soll die Mitarbeiter von schematischen Tätigkeiten entlasten und Arbeitskraft für kreatives Wirken freisetzen.
- Die Einbringung der graphischen Fähigkeiten in ein Programmsystem zur integrierten Behandlung von Entwurfsproblemen (REGENT = Rechnergestützter Entwurf) soll die frühzeitige Kopplung von Programmen zur Auslegung mit denen zu ihrer Darstellung in einem durchgehenden Prozeß mit gemeinsamer Datenbasis ermöglichen.

2. Eigenschaften bekannter graphischer Systeme

Die graphische Datenverarbeitung befaßt sich mit der Bereitstellung von Methoden zur Erzeugung, Manipulation und Ausgabe graphischer Information. Bei der Verarbeitung dieser Information muß immer der Schritt von der rechnerinternen Repräsentation der Daten, zur Interpretation dieser Werte als Koordinaten- oder Darstellungsangaben und schließlich zur Ansteuerung eines Ausgabemediums vollzogen werden. Im Gegensatz zur Darstellung numerischer Daten, für die die höheren Programmiersprachen geeignete Abbildungen bereitstellen, liegt dieser Vorgang weitgehend in der Verantwortung des Problemprogrammierers, dem dabei folgende graphischen Grundfunktionen zum Zeichnen:

- eines Vektors,
- einer Fläche (Bereiches) oder
- eines alphanumerischen Zeichens

zur Verfügung stehen. Diese Basisfähigkeiten werden auf den graphischen Ausgabegeräten:

- Linienplotter,
- Flächenplotter,
- Vektor- und
- Rasterdisplay

unterschiedlich realisiert.

Der Anstoß zur Ausführung einer Zeichenaufgabe erfolgt mittels Programm auf der Softwareebene. Die Auflösung dieser Anforderung in elementare Schritte des Zeichenstiftes oder -strahles wird je nach Gerätetyp von der Software- oder Hardware übernommen. In Abb. 2.1 sind die Realisierungsebenen für die Basisfähigkeiten den verschiedenen Gerätetypen zugeordnet. In den letzten Jahren ist dabei eine Tendenz, zur Übertragung auch komplexer Ausgabefunktionen auf die Hardware zu verzeichnen.

Während bei der Darstellung numerischer Werte sich der Einfluß der Hardware auf die Ausgabegeschwindigkeit beschränkt, wirken sich bei einem graphischen Gerät dessen Eigenschaften auf eine prinzipielle Realisierung von Fähigkeiten aus.

Ausgabeobjekt \ Gerätetyp		Linienplotter	Flächenplotter	Vektor-Display (Speicher-Wiederholung)	Raster-Display (Wiederholung)
Elemente	Vektor	Software (Hardware)	Software	Hardware	Hardware
	Fläche	-	Software	-	Hardware
	Zeichen	Software	Software	Hardware	Hardware
Komplexe	Bildelemente	Software	Software	Software	Software (Hardware)

Abb. 2.1: Realisationsebene für graphische Basisfähigkeiten

In der obigen Abbildung sind die graphischen Basisfunktionen festgehalten, die bei ihrer Ausführung mit weiterer Information bezüglich Sichtbarmachung, wie:

- Linienart,
- Strichstärke,
- Intensität,
- Farbe, usw.

versehen werden müssen. Auch hier schließt die Verwendung eines bestimmten Ausgabegerätes u. U. die sinnvolle Interpretation eines Darstellungsattributs aus.

Durch diese Unverträglichkeit der Hardwarekonzepte und - vor allem - durch die auf ihrer Basis entwickelten Softwarepakete der Gerätehersteller ist eine sinnvolle Verwendung graphischer Software mit einem hohen Maß an Übertragbarkeit auf andere Installationen bis heute nicht gelungen.

Die Realisierung graphischer Systeme muß sich deshalb auf ein Minimum an geräteabhängiger Grundsoftware abstützen.

In Ermangelung eines allgemein akzeptierten Konzeptes zum Angehen der graphischen Probleme findet sich in der Literatur eine Vielzahl von verschiedenartigen Ansätzen für graphische Systeme, um die obengenannten Basisfähigkeiten der graphischen Ausgabegeräte auch für den Problemprogrammierer nutzbar zu machen. Diese Ansätze lassen sich nach der in ihnen begangenen Art des Lösungsweges in zwei Gruppen aufteilen:

- Sprachen zur Beschreibung zwei- und dreidimensionaler Strichgraphik und
- Systeme zur Behandlung von räumlichen Objekten.

Im folgenden sollen die Eigenschaften der aus der Literatur bekannten Systeme für diese Hauptarbeitsgebiete der graphischen Datenverarbeitung betrachtet und hinsichtlich einer Aussage für die Realisierung eines neuen Entwurfes untersucht werden.

2.1 Sprachen graphischer Systeme für zwei- und dreidimensionale Strichgraphik

Jede Eingabe, die von einer Rechenanlage als Auftrag zur Erzeugung und Manipulation von abstrakten Objekten verstanden wird, kann als formale Sprache bezeichnet werden. Diese Sprachdefinition umfaßt die Bedienung der Rechnerkonsole ebenso wie die Formulierung und Eingabe von Problemen in einer höheren Programmiersprache.

Für die weiteren Überlegungen sollen jedoch nur die höheren Programmiersprachen Beachtung finden, da auf ihrer Ebene die im Bereich des Rechnergestützten Entwurfs (Computer Aided Design, CAD) anfallenden Probleme angegangen werden und sich hier die Notwendigkeit zur Nutzung graphischer Fähigkeiten ergibt.

Die Entwicklungen von intelligenten graphischen Terminals mit dem umfangreichen Befehlsvorrat eines speziellen graphischen Assemblers und den Möglichkeiten einer Display-File-Programmierung /83,106/ ist daher nicht Gegenstand der weiteren Betrachtungen.

Unterprogrammaufruf

Die älteste Möglichkeit zur Ansteuerung von graphischen Ausgabeeinheiten ist die Anwendung von Aufrufen eines Softwarepaketes der Gerätehersteller /33/. In allen höheren Programmiersprachen lassen sich Gruppen von Anweisungen zu Subroutinen bzw. Prozeduren zusammenfassen und für eine gemeinsame spätere Ausführung abspeichern. Diese Technik führt zu einer größeren Überschaubarkeit der Programme und zu höherer Modularität, da sich die Problemlösungen eines speziellen Anwendungsgebietes in separierten Programmpaketen realisieren lassen.

Während die Rechnerfirmen in besonderen Bibliotheken Routinen für bestimmte mathematische Standardaufgaben - z. B. trigonometrische Funktionen - bereitstellen, unterstützen die Hersteller graphischer Geräte die Anwendung ihrer Ausgabeeinheiten durch Programme zu deren Ansteuerung. Diesen Unterprogrammen werden die zur Ausführung der graphischen Ausgabeoperation erforderlichen Daten über ihre Argumentliste übergeben. Diese können dabei auch neben Konstanten, Variablen oder arithmetische Ausdrücke enthalten, so daß eine für die Ausführung des Aufrufes ausreichende Variabilität erreicht werden kann. Störend wirkt sich jedoch die starre Syntax derartiger

Aufrufe aus, da sie bei längeren Listen zu einer Unübersichtlichkeit führt, in der die Bedeutung des einzelnen Argumentes, insbesondere seine Position in der Liste, unsicher wird.

Eine Einprägsamkeit der Aufrufe muß aber dann gefordert werden, wenn der Anwender die graphische Software als flexibles Werkzeug zur Formulierung des graphischen Sachverhaltes nutzen soll.

Die zeichnerische Ausgabe einer Linie als Verbindung zweier Punkte wird durch die folgenden Anweisungen bei Anwendung eines Unterprogrammaufrufes bewirkt:

```
INTEGER PEN
REAL  XC(2), YC(2)
XC(1) = 1.
YC(1) = 3.
XC(2) = 5.
YC(2) = 7.
PEN = 2
CALL LINE (XC,YC,PEN)
```

Der Anwender muß bei diesem Vorgehen selbst die Zuordnung zwischen dem gewünschten Ausgabevorgang und der internen Repräsentation der Daten herstellen. D.h. er muß neben der Operation "Zeichne Linie" die Speicherung der Punktkoordinaten planen, während er zum Beispiel bei arithmetischen Operationen dieser Überlegung enthoben ist. Für die Speicherung arithmetischer Konstanten und Variablen kennt der Compiler die geeignete Speicherungsform.

Eine Programmiersprache für graphische Anwendungen muß den Objektumfang auf Punkte, Texte, Linienzüge usw. erweitern, damit ein Referieren der zugehörigen Variablennamen eine dem graphischen Gehalt des Objektes entsprechende Operation bewirkt. Das graphische Objekt enthält alle zu seiner Manipulation oder Ausgabe erforderlichen Informationen.

Einlesen von Konstanten

Moderne Betriebssysteme stellen für die Eingabe von Programmdaten verschiedene Medien bereit. Bei Programmen, die ein abgeschlossenes, fest einprogrammiertes Modell eines technischen Vorganges enthalten, werden diese Medien (Karten, Lochstreifen, Magnetband, Terminal usw.) zur Eingabe von Parametern benutzt, die die Eigenschaften von Komponenten des Systems an die zu modellierende Wirklichkeit anpassen.

Im Falle graphischer Anwendungen sind dies die Art der geometrischen Beziehung sowie die Attribute der geometrischen Objekte beeinflussende Angaben. Die folgenden tabellarischen Angaben könnten eine mögliche Eingabeform für die bereits oben verwendete Aufgabe "Zeichnen einer Linie" darstellen.

Identifikation	Element oder Operator	Daten
1	POINT	1. 3.
2	POINT	5. 7.
3	LINE	1 2

In dieser Tabelle werden einem graphischen System Anweisungen nur durch die formatgebundene Angabe von numerischen Konstanten und Zeichenketten erteilt. Die erste Spalte enthält die Angabe zur Identifikation des graphischen Grundelementes, das durch die folgenden Angaben in seinen Eigenschaften beschrieben wird. Die häufig angewendete Technik, z.B. verschiedene Operationen durch unterschiedliche Kartenarten zu kennzeichnen, ist eine Variante der hier beschriebenen Vorgehensweise.

Durch den Zwang zur Wahrung von Spaltenrestriktionen und die Beschränkung auf Konstanten ist eine Programmierung von graphischen Aufgaben mittels dieser "Sprache" ein umständliches und zeitaufwendiges Unterfangen. Zudem verbietet diese Eingabeform die Verwendung von arithmetischen Ausdrücken, von Namen und von Unterprogrammen, die das Anwendungsspektrum und die Handhabbarkeit einer Programmiersprache maßgeblich bestimmen.

In beiden bisher beschriebenen Methoden zur Erzeugung graphischer Ausgabe wird eine Auswahl numerischer Daten als Attribute graphischer Objekte interpretiert und zu deren zeichnerischer Darstellung genutzt. Die in Abb. 2.2 zusammengestellten bekanntesten Entwürfe bzw. Realisierungen graphischer Sprachen gehen weiter, indem sie versuchen, graphische Objekte in die Sprache einzubeziehen.

Die von den Autoren gewählten Wege lassen sich nach verschiedenen Eigenschaften gliedern, die wiederum unter zwei Hauptgesichtspunkten zusammengefaßt werden können:

- graphische Eigenschaften und
- Methode der Sprachrealisierung.

In den beiden nächsten Abschnitten sollen diese Systeme unter diesen Gesichtspunkten betrachtet werden.

2.1.1 Graphische Eigenschaften

Die Hauptaufgabe einer graphischen Sprache liegt in ihrer Fähigkeit, die der Zeichentätigkeit zugrundeliegenden Vorgänge in abstrakter Weise formulierbar zu machen. Von beherrschendem Einfluß sind dabei die Begriffe, die

- geometrische Merkmale und
- Darstellungsattribute

betreffen.

Geometrische Merkmale

Zu den geometrischen Merkmalen dieser Sprachen gehört ihr Vorrat an:

- geometrischen Objekten und
- geometrischen Operationen.

Für die graphische Programmierung sind in den meisten Systemen an Objekten nur die graphischen Primitive

- Punkt (PU)
- Polygon (PO) und
- Text (TE)

nutzbar, wobei in der Realisierung dieser Grundfähigkeiten jedoch unterschiedliche Konzepte zum Tragen kommen.

Während bei /79/,/131/,/162/,/58/ und /47/ hinter diesen Objekten auch Variable des entsprechenden Datentypes stehen, die als Ergebnis vorangegangener Operationen interpretierbare geometrische Daten enthalten, verstehen die anderen Systeme die Operationen POINT, LINE, usw. nur als aktuellen Auftrag einer graphischen Funktion zur Ansteuerung eines Zeichengerätes /69/,/111/,/105/,/16/,/107/,/133/.

Für eine sinnvolle graphische Programmierung ist jedoch die Beschreibung und Speicherung mit abstrakten Objekten, die den geometrischen Informationsgehalt eines Bildes repräsentieren, unerläßlich. Bei graphischen Sprachen gelten daher die gleichen Anforderungen, auf denen die Entwicklung von höheren Programmiersprachen für mathematische Anwendungen beruht. Dort werden die Ergebnisse arithmetischer Operationen, die aus der Anwendung von Basisoperationen oder deren Zusammenwirken in komplizierten Statements oder

Sprache	Autor	Forschungs-einrichtung (Land)	Jahr	Graphische Eigenschaften					Spracheigenschaften (Abk. s. Abb. 2.3)			Daten-struktur	Literatur
				Objekte	Operation	Dimens.	Darst. Attr.	Inter-aktiv	Art der formalen Spr. u. ihrer Realisierung	Technik (Basis)	semantische Routinen		
	Kulsrud	Yale Univ. (USA)	'67	PU, PO	EP, LT	2	-	-	P, EG	C(ASS)	FORTRAN	-	79
GRAF	Harwitz, Citron	IBM, LA (USA)	'67	-	EP	2	-	(x)	P,SE	PC(FORTRAN)	FORTRAN	-	69
DRAWL 70	Herzog	(USA)	'70			2	-					-	67
GRAFTRAN	Eklins, Wolf	(USA)	'72			2	-		P, SE	PC(FORTRAN)	FORTRAN	-	44
GPDL	Notley	ICL (GB)	'70	-	EP, LT	3	-	-	(P), EG	C	ASSEMBLER	-	111
	Anderson, Forber	Rand Corp. (USA)	'70	-	EP	2	-	(x)	P, SE	PC(PL/1)	PL/1	-	4
EULER-G	Newman	Univ. of Utah (USA)	'70	-	EP, LT	(3)	-	(x)	P, SE	ER	EULER	-	105
METAVISU	Boulier, e.a.	Rocquencourt (F)	'71	-	GB,LT	2	-	x	P, SE	PC(PL/1)	PL/1	Assoz.	16
LOGO-G	Newman	Queen Mary C. (GB)	'71	-	EP	2	-	-	P, SE	ER(LOGO)	LOGO	-	107
GPL/1	Smith	Boeing (USA)	'71	PU, PO, TE	EP, LT	3	x	(x)	P, SE	PC(PL/1)	PL/1	-	131
	Williams	NY Univ. (USA)	'72	PU, PO, u.a.	EP, LT	3	-	(x)	P, EG	PC(FORTRAN)	FORTRAN	Hierarch.	162
APL-G	Giloi, Encarnacao	TU Berlin (D)	'72	PU, PO, TE	EP, GB, LT	(3)	x	x	P, SE	ER(APL)	APL, L^4	Assoz.	58
L	Nake, e.a.	Univ. of Brit. Col. (CDN)	'72	-	EP	(2)	-	x	P, EG	PC (XPL, FORTRAN)	FORTRAN	-	110
GRAPL/1	Soop	IBM (S)	'72	-	EP	2	-	(x)	P, SE	PC(PL/1)	PL/1	-	133
GETAM	Brewer, Cordes	LSU (USA)	'73	(PO)	LT	3	-	x	K, EG	DI(ICES,CDL)	ICETRAN	-	32
GRAPHIC	Enderle, Schuster, e.a.	GFK (D)	'73	PU, PO, TE u.a.	EP, GB, LT	2	x	(x)	K(P), EG	DI(ICES,CDL)	ICETRAN	Assoz.	47
EUCLID	Mountford, e.a.	D-A Comp. Serv. (GB)	'73	-	GB	2	x	-	EG	C	FORTRAN	-	102
GEOLAN		MBB (D)	'75	PU, PO, u.a.	EP, GB, LT	3	-	-	K, EG		FORTRAN	-	103
GIPSY	Schuster	GFK (D)	'76	PU, PO, TE u.a.	EP, GB, LT	3	x	(x)	P,SE	DS(REGENT,PLS)	PLR	List	

Abb. 2.2: Vergleich bekannter graphischer Sprachen

PU - Punkt
PO - Polygon
TE - Text
EP - Erzeugung graphischer Primitive
GB - Ausführen geometrischer Berechnungen
LT - Lineartransformationen

Prozeduren resultieren, in Speicherplätzen, denen durch Datentypvereinbarung eine angepaßte Struktur zu eigen ist, abgelegt und einem späteren Aufruf zugänglich gemacht.

Diese Form der Abspeicherung ist eine wichtige Voraussetzung für die algorithmische Behandlung graphischer Probleme und eine namentliche Identifikation der Objekte.

In /47/ werden neben den obengenannten Primitiven z.B. auch Objekte für Kreisbögen, Kreise, Koordinatenachsen und Kurven in Problemkoordinaten angeboten, um so auch komplexere Aufgaben durch die Verwendung von zum Sprachumfang gehörenden Begriffen beschreibbar zu machen.

Bei Williams /162/ ist die Sprache so angelegt, daß sie mit eigenen Mitteln um neue Objekte erweiterbar ist.

Neben dem Vorrat an Objekten bestimmt die Auswahl an graphischen Operationen die Mächtigkeit einer graphischen Sprache. Nach dem oben genannten Unterschied im Konzept entspricht bei /69/,/111/,/16/,/107/ und /133/ die Anwendung einer Operation der gleichzeitigen Ausführung der angestoßenen Zeichenbewegung. In denjenigen Systemen, die graphische Objekte kennen, sind die Ergebnisse von Operationen abspeicherbar. Die graphischen Operationen erfüllen dabei folgende Aufgaben:

- Erzeugung von graphischen Primitiven (EP),
- Ausführen geometrischer Berechnungen (GB),
- Lineartransformationen (LT).

Für die in den Vergleich, der alle wichtigen Veröffentlichungen auf diesem Gebiet umfaßt, einbezogenen Sprachen gilt, daß sie nur die Behandlung der Probleme zwei- und/oder dreidimensionaler Strichgraphik erlauben.

Darstellungsattribute

Neben den geometrischen Angaben, die den Typ und die Lage eines graphischen Elementes betreffen, sind Attribute zu seiner Darstellung anzugeben. Diese Angaben erleichtern die Interpretation eines Bildes, wenn sie eine Bildinterpretation durch Nutzung der Eigenheiten des menschlichen Sehens /98/ oder der angelernten Fähigkeiten der Auslegung technischer Darstellungsregeln erlauben.

Von den verglichenen Sprachen bieten nur /131/,/58/,/47/ und /102/ diese Möglichkeit zur Verwendung von Darstellungshilfen.

Die graphischen Eigenschaften einer Objektdarstellung sind dabei abhängig vom:

- System- und
- Objektzustand.

Erst eine Abbildung eines abstrakten, gespeicherten Objektes unter Wahrung der im System- und Objektzustand befindlichen Attribute führt zur sichtbaren graphischen Information.

2.1.2 Methoden der Sprachrealisierung

Neben graphischen Eigenschaften bestimmen die Möglichkeiten zur algorithmischen Formulierung von Problemen die Einsetzbarkeit einer graphischen Sprache. Die Wege, auf denen die hier betrachteten Systeme ihre Fähigkeiten bereitstellen, sind in Abb. 2.3 in einem Graphen verdeutlicht.

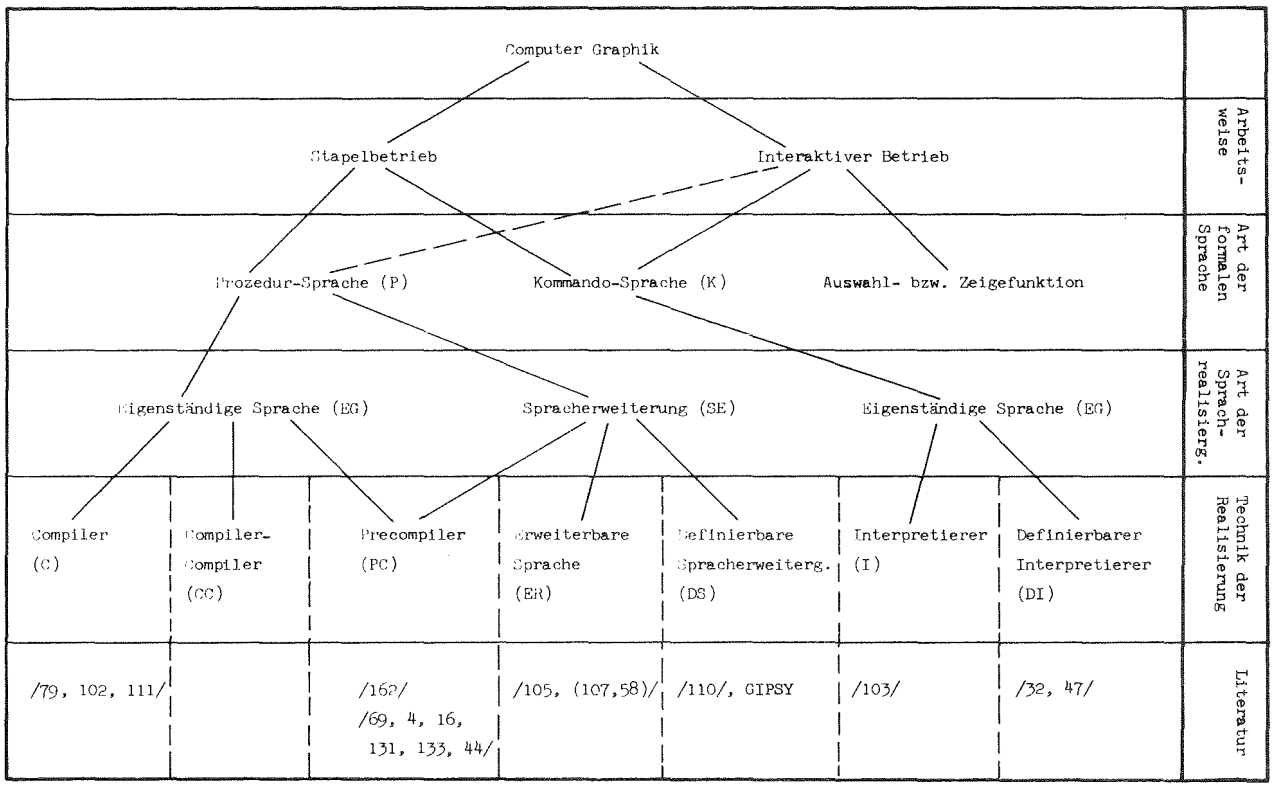


Abb. 2.3: Methoden der Sprachrealisierung graphischer Sprachen

Vom Typ unterscheidet man

- Prozedur (P)- und
- Kommandosprachen (K),

die durch

- den Entwurf einer eigenständigen Sprache (EG) oder
- die Erweiterung einer verfügbaren höheren Programmiersprache (SE)

realisiert sind.

Als eigenständig bei den prozeduralen Sprachen können nur die Vorschläge von Kulsrud /79/ und Notley /111/ bezeichnet werden, da sie auf einer unabhängig definierten Syntax, die sich auf die graphischen Anforderungen konzentriert, basieren. Ihre Fähigkeiten z.B. hinsichtlich Programmablaufsteuerung oder Ein/Ausgabeoperationen sind jedoch beschränkt.

Für eine problemangepaßte graphische Sprache müssen jedoch auch diese Merkmale moderner höherer Programmiersprachen erfüllt sein. Da die Entwicklung einer solchen Sprache wegen des auf graphische Anwendungen hin noch erweiterten Umfangs nur mit großem Aufwand - bei PL/1 einige hundert Mannjahre /112/ - durchführbar wäre, gewinnen Spracherweiterungen zunehmende Bedeutung /76, 115, 130, 146/.

Bei den untersuchten Systemen wurden die graphischen Fähigkeiten hinzugefügt durch:

- eine Erweiterung des verfügbaren Compilers /107, 58/,
- die Anwendung eines Vorübersetzprogrammes (Precompiler, PC) /69, 4, 16, 131, 133/ oder
- die Verwendung einer erweiterbaren Sprache (ER) /105/.

Dabei werden APL /58/, LOGO /107/, EULER /105/, FORTRAN /69, 110/ und PL/1 /16, 4, 131, 133/ als Gastgebersprache ('host language') benutzt.

Die Mehrzahl der Spracherweiterungen basiert auf PL/1, da diese Sprache wichtige Eigenschaften bereits in ihrem Sprachumfang anbietet, die für graphische Anwendungen nützlich - wenn nicht unabdingbar - sind, wie die Möglichkeiten:

- zur Stringmanipulation (Bit und Character),
- zum Aufbau von Datenstrukturen,
- zum Interrupt-handling und
- zur Erweiterung um neue Sprachelemente.

Ein wichtiger Gesichtspunkt für die Implementierung und vor allem für die Verbreitung einer graphischen Sprache ist die Verfügbarkeit und Zuverlässigkeit des Compilers der Basissprache.

2.2 Systeme zur Behandlung räumlicher Objekte

In diesem Abschnitt werden diejenigen Systeme zusammengestellt und verglichen, in denen räumliche Objekte mit dem Rechner behandelt werden. In Abb. 2.4 und Abb. 2.5 sind die untersuchten Systeme chronologisch nach Veröffentlichungsjahr aufgeführt.

Die Autoren konzentrieren sich fast ausschließlich - die Ausnahmen /37/ und /154/ bestätigen dies nur - auf die geometrischen Probleme bei der Darstellung der Objekte auf einer zweidimensionalen Zeichenebene und dabei vor allem auf die Erarbeitung von Sichtbarkeitskriterien.

Die Eingabeform zur Formulierung der graphischen Aufgabe kann nur bei Engeli /37/ und Takasawa, e.a./154/als graphische Programmiersprache bezeichnet werden. Bei vielen Autoren fehlen die Angaben zu den Möglichkeiten zum Ansprechen der Systemfähigkeiten völlig. Nach einigen Hinweisen kann vermutet werden, daß es sich wie bei /1/, /59/, /38/ und /96/ um numerierte Listen von Punkten, Kanten und Flächen handelt, aus denen sich die Eingabe vor allem für Polyeder aufbaut.

Einige Systeme erlauben die interaktive Eingabe am Terminal /127, 38, 78/, während Roberts /122/ seine Eingabe einer Photographie des Objektes entnimmt. Bei der Erstellung der Eingabe von graphischen Flächen kommen häufig Digitalisierungshilfen zum Einsatz, da die zur Bestimmung der Flächenparameter erforderliche Datenmenge sehr umfangreich ist /11/.

Die Aufteilung der graphischen Systeme in solche mit graphischen Sprachen als Eingabe und solchen ohne diese Formulierungshilfen ist damit praktisch gleichbedeutend mit der Trennung zwischen Systemen für graphische Strichinformation und denjenigen zur Behandlung räumlicher Objekte.

Die geringe Bedeutung, die bisher der Erleichterung der Eingabe beigemessen wurde, muß als Haupthindernis für die Anwendung und Verbreitung dieser Systeme angesehen werden.

Im Hinblick auf die Ziele dieser Arbeit werden die Systeme für Quadriken im folgenden getrennt behandelt.

Autor (System)	Forschungs-einrichtung (Land)	Jahr	Flächen	Flächendarst. intern	Flächendarst. extern	Sichtbark.algor. Entsch.raum	Sichtbark.algor. Prinzip	Implement.-Sprache	Eingabe-form	Interaktiv	Literatur
Roberts	MIT (USA)	'63	PE	FB	LD	3D	PT	ASSEMBLER	Photo	-	122
Sutherland (SKETCHPAD)	MIT (USA)	'64	PE	-	LD	-	-	ASSEMBLER	LP	x	127
Coons	MIT (USA)	'67	GF	PA	RL	-	-		Fläche (P)		26
Loutrel	NY Univ. (USA)	'67	PE	GE	LD	2D	KT	FORTRAN		-	90
Appel	IBM, NY (USA)	'67	PE	GE	LD	2D	QU	FORTRAN	Liste (PKF)	-	1
Wylie	Univ. of Utah (USA)	'67	GF	GE	HT	2D	BU				165
Ricci	CNEN, Bologna (I)	'68	AF	FB	RL	2D	RA	FORTRAN	Funktionen	x	123
Kubert, e.a.	Aerospace, Ca. (USA)	'68	AF	FB	RL	3D	PT	(FORTRAN)	Funktion	-	81
Warnock	Univ. of Utah (USA)	'69	GF	GE	RL	2D	BU	(FORTRAN)		-	168
Galimberti, e.a.	Polyt. Milano (I)	'69	PE	GE	LD	(3D)	KT	FORTRAN	P-K-F	-	59
Watkins, G.S.	Univ. of Utah (USA)	'70	GF	GE	HT	2D	BU	(FORTRAN)			169
Bouknight	Univ. of Illinois (USA)	'70	GF	GE	HT	2D	BU	(FORTRAN)			7
Kamiuchi	Kyoto Univ. (J)	'70	PE	GE	LD		PT	-	-	-	84
Encarnacao (PRADIS)	TU Berlin (D)	'70	PE,GF	GE,PA	LD,RL	3D,2D	KT,BU	FORTRAN	PKF, Fkt., Fl.	x	38
Bézier (UNISURF)	Renault (F)	'71	GF	BK	RL	-	-		P		11
Forrest	CAD, Camb. (GB)	'71	GF	PA	RL			FORTRAN	P		50
Mc Grath	TRW, Ca. (USA)	'71	PE	GE	LD	2D	KT	FORTRAN	Namelist(PK)	-	96
Gouroud	Univ. of Utah (USA)	'71	GF		HT	3D		FORTRAN	P		54
Kurth (COMPAC)	TU Berlin (D)	'71	PE	GE	LD	-	-	FORTRAN	Kommandos	-	80
Nakamae,e.a.	Hiroshima (J)	'72	PE	GE	LD	2D	KT				109
Takasawa, e.a.	Univ. of Tokyo (J)	'72	PE	GE	LD			FORTRAN	GML		154
Williamson	Tracor, Texas (USA)	'72	GF	PR	RL	2D	RA	FORTRAN	Call	-	161
Furukawa	Yamanshi Univ. (J)	'73	PE	GE	LD	3D	PT	FORTRAN		-	52
Engeli (EUKLID)	Fides Treuh. (CH)	'73	PE,GF	GE,BK	LD,RL				EUKLID	-	37
Kira(IMAGES)	NHK (J)	'74	PE	GE	LD	2D	KT	FORTRAN(ASS)		x	78
Watkins, S.L.	Univ. of Texas (USA)	'74	GF	PR	RL	2D	RA	FORTRAN	Call	-	170
Herold (PROREN)	Univ. Bochum(D)	'74	PE	GE	LD	(2D)	KT	FORTRAN	Kommandos		71
Wright	NCAR (USA)	'74	GF	PR	RL	2D	RA	FORTRAN	Call	-	171
Schuster (GIPSY)	GFK (D)	'76	PE,AF, GF	FB,PR	LD,RL	3D,2D	(KT,PT), RA	REGENT (PL/1)	GIPSY	(x)	

Abb. 2.4: Vergleich bekannter Systeme zur Behandlung räumlicher Objekte (ohne Flächen 2. Ordnung)

Flächenformen

Der Vorrat an Flächenformen zur Beschreibung der räumlichen Objekte umfaßt:

- Ebenen (PE)
  /90, 1, 59, 84, 38, 80, 109, 154, 52, 37, 78, 71/,
- graphische Flächen (GF)
  /165, 168, 169, 7, 38, 11, 161, 37, 170, 171/ und
- analytische Flächen (AF)
  /123, 81/.

Während die Ebene ausschließlich zum Aufbau von Körpern, d.h. abgeschlossenen Volumina (Polyeder, PE), herangezogen wird, beschreiben die graphischen Flächen nur Objekte mit Eigenausdehnung 2. Bei den analytischen Flächen (ohne Quadriken) betrachtet Ricci /123/ Körper, während Kuberth, e.a. /81/ Flächen untersuchen.

Flächendarstellung

Für die rechnerinterne Repräsentation der graphischen Information wurden folgende Verfahren angewendet:

- Speicherung einer Struktur primitiver Grundelemente, wie Punkte, Polygone, Ebenen usw. (GE),
- Zerlegung der Fläche in Pflaster mit geeigneter Parameterdarstellung zur Generierung beschreibender Rasterlinien (PA, BK, PF) und
- exakte Beschreibung der Fläche in impliziter, expliziter oder parametrischer Form (FB).

Die Mehrzahl der Systeme beschränkt sich auf die Behandlung von Polyedern, da diese sich einfach mittels primitiver Grundelemente einschließen lassen, aber auch für graphische Flächen wird in /165/, /168/, /169/ und / 7/ diese Methode verwendet.

Die Zerlegung von graphischen Flächen ('free form surfaces') in parametrisch beschreibbare Pflaster ('patches', PA) wurde erstmals von Coons /26/ durchgeführt und von weiteren Autoren übernommen bzw. verbessert /38, 50, 8, 9/. Bezier /11/ und Engeli /37/ spannen in ein Punkteraster Polynome (BK) mit vorgebbaren Eigenschaften.

Die Möglichkeit, graphische Flächen, die nur durch ein Wertfeld der in der Fläche liegenden Punkte festgelegt sind, perspektivisch unter Anwendung eines Sichtbarkeitskriteriums darzustellen, bieten /161/, /170/ und /171/.

Die externe Flächendarstellung ist neben der gewählten internen Darstellung auch von den Möglichkeiten der graphischen Hardware abhängig. Von den Möglichkeiten der Darstellung durch

- Linien und
- Schattierungen

läßt sich letztere nur auf Flächenplottern und Rasterdisplays realisieren (vergl. Abb. 2.1).

Bei der Liniendarstellung unterscheiden wir wiederum zwei Arten von Linien:

- solche, die beim Schnitt von Flächen als Durchdringungskurven und bei Betrachtung aus bestimmter Richtung als Umrißkurven entstehen (LD) und
- solche, die als Kontur- oder allgemein Rasterlinien auf einer Fläche erzeugt werden und die Veranschaulichung der Fläche unterstützen (RL).

Soweit als Darstellungshilfe nicht Schattierungen gewählt wurden /165, 169, 7/, benutzen alle Systeme für graphische und analytische Flächen die Veranschaulichung mittels Rasterlinien.

Sichtbarkeitsalgorithmen

Der Schwerpunkt des Interesses bei allen verglichenen Systemen liegt beim Entwurf und der Implementierung eines Sichtbarkeits- (Hiddenline-) Algorithmus. Da die Bestimmung der Sichtbarkeit einen entscheidenden Beitrag zur Effektivität eines Systems liefert, werden immer wieder neue Verfahren ersonnen, die im Vergleich mit existierenden ihre Fähigkeit beweisen müssen.

Nach Loutrel /90/, Encarnacao /40/ und Demic /35/ sind vor allem Vergleiche von Sutherland e. a. /129/ und Becker /20/ zu nennen, die zehn bzw. zwanzig Verfahren einander gegenüberstellen. Wichtigste kennzeichnende Merkmale eines Verfahrens sind dabei:

- der Entscheidungsraum und
- das -prinzip der Sichtbarkeitsbestimmung.

Hinsichtlich des Entscheidungsraumes werden Objekt- (3D) und Bildraum (2D) unterschieden, obwohl auch die letzteren Verfahren auf eine Zusatzinformation über die relative räumliche Lage zweier im Bild zusammenfallender Punkte zur Entscheidung nicht verzichten können.

Nach dem Entscheidungsprinzip ergibt sich folgende Einteilung:

- Punktetest (PT),
- Kantentest (KT),
- Bildunterteilung (BU) und
- Sichtbarkeitsfeldberandung (RA).

Zu den Punktetests gehört das erste implementierte Verfahren zur Sichtbarkeitsbestimmung von Polyedern von Roberts /122/, bei dem die Sehstrahlen vom Projektionszentrum zum zu untersuchenden Punkt mit den Flächen der konvexen Teilvolumina zum Schnitt gebracht werden. Liegt einer dieser Schnittpunkte zwischen Zentrum und dem getesteten Kurvenpunkt, so ist dieser unsichtbar. Der Punktetest, bei dem ausschließlich im Raum über die Sichtbarkeit entschieden wird, zeichnet sich durch seine universelle Verwendbarkeit und durch seine einfache Realisierbarkeit aus, kann aber bei komplexen Objekten zu nicht akzeptablen Rechenzeiten führen.

Wiederum für Polyeder hat Appel /1/ einen Algorithmus zur Bestimmung der quantitativen Verdeckung ('quantitative invisibility') eingeführt. Dabei wird davon ausgegangen, daß beim scheinbaren Schnitt zweier Polyederkanten im Bild diejenige, die hinter eine Kante läuft, den Grad ihrer Unsichtbarkeit entsprechend der Anzahl sie nun mehr verdeckender Ebenen um zwei erhöht oder im umgekehrten Fall vermindert. Dieses Verfahren, das wie das folgende von Loutrel /90/ zu den Kantentests zu rechnen ist, arbeitet vorwiegend im Bildraum.

Loutrel führt zur Beschleunigung seines Algorithmus eine sogenannte Kantenklassifikation ('edge classification') ein, die die Polyederkanten in unsichtbare und potentiell sichtbare je nach Orientierung der sie bildenden Ebenen im Raum vorsortiert. Im nachfolgenden Test für die potentiell sichtbaren Kanten wird die Ordnung der Unsichtbarkeit ('order of invisibility') festgelegt, wobei überprüft wird, ob ein Punkt P_o der Kante durch eine Frontebene vom Projektionszentrum getrennt wird und ob der Bildpunkt P'_o innerhalb der Projektion der verdeckenden Fläche, d.h. des sie umrandenden Polygonzuges liegt. Solange eine Kante keinen scheinbaren Schnittpunkt mit einer Konturkante besitzt, bleibt die Ordnung konstant. Beim Auftreten von Schnittpunkten muß ähnlich vorgegangen werden wie bei der quantitativen Verdeckung.

Die Algorithmen zur Bildunterteilung /165, 168, 169, 7, 38/ versuchen den Aufwand zur Sichtbarkeitsbestimmung dadurch zu reduzieren, daß sie durch fortgesetzte Viertelung des Bildes ('subdivision') oder durch das Überziehen des Bildes mit Prüflinien ('scan lines') die zum Sichtbarkeitstest erforderliche Anzahl von Flächenelementen oder Rasterpunkten verringern. Herangezogen wurde dieses Verfahren vorwiegend bei der Darstellung von graphischen Flächen, die durch ebene Grundelemente approximierend bedeckt waren, mittels Schattierung ('shaded pictures').

Diejenigen Verfahren, die auf einer rein zweidimensionalen Verwaltung der Grenzen eines Sichtbarkeitsfeldes (Maske) beruhen, haben für einen allgemeinen Einsatz keine Bedeutung, da ihre Darstellung mittels Kontur- oder Rasterlinien dem Anwender die Angaben über die räumliche Zuordnung überläßt /161, 170, 171/.

Dieser obige Vergleich bezieht sich auf die entscheidenden Merkmale der bisher implementierten Hiddenline-Algorithmen, die auch für die in Abb. 2.5 zusammengestellten Systeme von Bedeutung sind. Die vollständige Zuordnung aller Verfahren kann anhand von Abb. 2.4 vorgenommen werden.

Implementierungseigenschaften

Zu den Implementierungseigenschaften zählen:

- die Eingabeform,
- die Datenstruktur und
- die Implementierungssprache.

Die zu diesen Punkten in der Literatur enthaltenen Angaben sind in der Regel spärlich. Die Eingabeformen wurden bereits zu Beginn dieses Abschnittes angesprochen. Es bleibt festzuhalten, daß außer zur Beschreibung von Polyedern bei /37/ und /154/ keine Eingabesprache existiert. Als Basis für die Implementierung wurde bei fast allen Systemen FORTRAN gewählt. Ein Hauptgrund hierfür ist sicher, daß zum Zeitpunkt der Realisierung vieler Systeme FORTRAN die einzige Sprache mit großer Verbreitung und hinreichender Standardisierung war, um eine Übertragung des Systems zu gewährleisten.

Auf die Datenstrukturen wird unter 2.4 noch gesondert eingegangen werden.

2.3 Systeme zur Behandlung räumlicher Objekte mit Flächen 2. Ordnung.

Der Schwerpunkt der Anwendungen graphischer Systeme liegt bisher bei der Beschreibung von Polyedern und sogenannten graphischen Flächen (vergl. Abschnitt 2.2). Die Oberflächen technischer - vor allem spangebend bearbeiteter - Werkstücke setzen sich neben Ebenen meist aus Formen kugeliger, zylindrischer oder kegeliger Gestalt zusammen. Diese Flächen fallen unter die Flächen 2. Ordnung (Quadriken, 'quadric surfaces').

Durch Luh und Krolak /88/ und Weiss /167/ wurden diese Flächen etwa gleichzeitig in die Computergraphik eingeführt.
Bis heute wurden folgende Arbeiten zu diesem Thema bekannt: Luh, Krolak (1965) /88/, Weiss (1966) /167/, Woon (1970) /164/, Wenz (1973) /142/, Braid (1975) /22/ und Becker (1975) /20/.

Diese Systeme werden in Abb. 2.5 hinsichtlich

- zugelassener Flächenformen,
- Lösungsweg der Kantenbestimmung,
- Sichtbarkeitsalgorithmus,
- Eingabeformen und
- Implementierung

verglichen.

Flächenformen

Die zulässigen Flächenformen sind bei /167/, /164/, /142/ und /20/ alle Flächen 2. Ordnung. Luh und Krolak /88/ beschränken sich auf die Auswahl von Kugel, Zylinder und Kegel. Braid /22/ erlaubt in seinem System den Körperaufbau aus sechs einfachen Grundformen, wobei neben Ebenen eine begrenzende Oberfläche zylindrisch sein darf. Das Programm SPACEPLOT /142/ kennt neben Quadriken den Torus als zusätzliche Fläche höherer Ordnung.

Der umfangreichste Flächenkatalog steht bei Becker /20/ zur Verfügung, da dort diejenigen Flächen verarbeitet werden können, die durch die Bewegung eines Kegelschnittes oder einer Geraden auf einer und um eine Raumkurve entstehen. Hierbei sind die Flächen 2. Ordnung nur eine Untermenge der darstellbaren verallgemeinerten Röhrenflächen, Kugelhüll- oder echten Röhrenflächen und Regelflächen.

System	Autor	Forschungseinrichtung	Jahr	zugelassene Flächen				interne Darstellung	Lösungsweg		Sichtbarkeitsalgorithm.		Implementierungssprache	Eingabeform	Anzahl der Flächen	Literatur
				Polyeder	Quadrik	höhere + transzendente	spezielle Endbegrenzung		Schnitt	Umriß	Entscheidungsraum	Prinzip				
	Luk, Krolak	IBM (USA)	'65	x	(Kugel, Zylinder, Kegel)	-	-	Koeff.: a_{ik}	numerisch	numerisch (nur Normalproj. in (x_2x_3)-Ebene)	3D	PT	FORTRAN	Kommandosprache	fest	88
BEVISION	Weiss	BELL (USA)	'66	x	x	-	x	Koeff.: a_{ik}	numerisch $Q_1(\underline{X})=Q_2(\underline{X})$	Bestimmung mit Tangentialebenen	3D	PT	FORTRAN	Tabellen mit Koeffizienten der Flächengleichg. (a_{ik})	fest	167
QUADRAW	Woon	N.Y. Univ. (USA)	'70	-	x	-	x	Koeff.: a_{ik}	numerisch	numerisch (Bestimmung der Bildkurve)	(2D)	KT (edge classification)	FORTRAN	Tabellen mit Flächendaten	fest	164
SPACEPLOT	Wenz, e.a.	WENZ, FfM. (BRD)	'73	x	x	(x)	(x)	Koeff.: a_{ik}	numerisch	Behandlung als Schnitt mit Polarebene	3D	PT	FORTRAN	Tabellen mit Punkt- und Flächendaten	22Fl. je Teilk, 100 Teilkörper 500 Flächen 2000 Punkte	142
BUILD (BILDFORT)	Braid	CAD Centre (GB)	'75	x	(Zylinder)	-	(x)	Parameter	nur senkr. Schnitt Zyl.-Ebene	keine Umrißbest. (4 Mantellinien)	?	?	FORTRAN	Kommandosprache (FORTRAN-Aufrufe)	fest	22
GIULIA	Becker	TH Aachen (BRD)	'75	-	x	x	x	Koeff.: a_{ik} (Erzeug., Leitkurve)	numerisch Erzeugende in Hilfsebene	numerisch	(2D)	KT (quantitative invisibility)	FORTRAN	Kommandosprache	(5)	20
GIPSY	Schuster	GFK, Karlsr. (BRD)	'76	x	(Kugel, Zylinder, Kegel)	-	-	geometrisch notwendige Angaben	analytisch	analytisch	3D	Kombinierter KT (edge class.) und PT	REGENT (PLR, PLS)	GIPSY-graph. Erweiterg. von PL/1	variabel	

Abb. 2.5: Vergleich bekannter Systeme zur Behandlung von Flächen 2. Ordnung

Diese Art der internen Darstellung ermöglicht z.B. die Behandlung eines geraden Kreiskegels durch vier verschiedene Ansätze. Dies sind die drei oben genannten und die gleichungsmäßige Abspeicherung durch die Koeffizienten der quadratischen Form. Für die erforderlichen Schnitte von Flächen untereinander kann dann die günstigste Repräsentation gewählt werden.

Lösungswege der Kantenbestimmung

Für die zur Liniendarstellung der Objekte notwendigen Bestimmungen von

- Durchdringungs- und
- Umrißkanten

verwenden alle im Vergleich erfaßten Systeme numerische Methoden. Bei /88/, /167/, /164/ und /142/ werden numerisch gemeinsame Punkte der beteiligten Quadriken gesucht, während /20/ auf der Grundlage der erzeugenden Kegelschnitte den Durchdringungskurvenverlauf bestimmt. Dabei sind bei Becker /20/ und bei Woon /164/ keine Schnitte zwischen Ebenen erlaubt.

Diese numerische Ermittlung der Punkte auf den Durchdringungskurven ist mit großem Rechenaufwand verbunden. Die dabei auftretenden numerischen Probleme bedingen teilweise den Übergang auf doppelte Rechengenauigkeit /142/.

Alle Verfahren müssen mit zusätzlichem Aufwand die Zuordnung der maximal vier Raumpunkte zu den Kurvenästen der entstehenden Raumkurve 4. Ordnung festlegen.

Im vorliegenden System wurden noch zu erläuternde analytische Methoden entwickelt, die eine geschlossene Behandlung des Durchdringungsproblems erlauben. Die Raumkurve entsteht in zusammenhängender Punktefolge als Ergebnis einer effektiven Berechnung in Abhängigkeit eines Parameters.

Braid /22/ bleibt mit den in seinem System realisierten Fähigkeiten hinter denen der übrigen weit zurück, da er keine echte Schnitt- und Umrißkurvenberechnung unterstützt. Seine Arbeit wurde mit in den Vergleich aufgenommen, da er großen Wert auf die Beschreibungsmöglichkeiten für Körper, die nicht nur von Ebenen eingeschlossen sind, legt.

Alle anderen Programme unterstützen eine betrachtungsabhängige Berechnung von Umrissen, die bei /88/ aber auf die Normalprojektion in die (x_2, x_3)-Ebene beschränkt ist. Die übrigen ermitteln den Umriß alle auf unterschiedliche Arten:

- Bestimmungen von Punkten in den Tangentialebenen /167/,
- Behandlung wie Durchdringungskurve nach Ermittlung der Polarebene der Fläche zum aktuellen Projektionszentrum /142/ und
- direkte Berechnung der Bildkurve nach Bestimmung der Polarebene wie vorgenannt /164/.

Auch bei der Bestimmung der Umrißkurven wird im vorliegenden System die Möglichkeit zur analytischen Berechnung genutzt. Der rechnerische Aufwand reduziert sich dann auf die Parametervariation entlang einer vorgegebenen Raumkurve.

Sichtbarkeitsalgorithmen

Die Charakteristika der Sichtbarkeitsalgorithmen wurden unter 2.1 untersucht, so daß hier eine Zuordnung zu den dort klassifizierten Verfahren ausreichen soll. In /88/, /167/ und /142/ wurden reine Punktetests realisiert, die ihre Entscheidungen im Objektraum vornehmen.

Woon /164/ überträgt den von Loutrel /90/ für Polyeder gefundenen Algorithmus der Kantenklassifikation auf die Flächen 2. Ordnung, während Becker /20/ Appels Methode /1/ der quantitativen Verdeckung auf seine Flächentypen anpaßt, wobei aber durch die Tatsache, daß Kanten nicht durch eine Gerade, sondern durch viele kleine Kurvensegmente gebildet werden, ein ungleich höherer Aufwand in die Bestimmung der scheinbaren Schnittpunkte von Körperkanten in der Bildebene fließt.

Solange diese Schnittpunkte über den Schnitt zweier aus vielen Segmenten bestehener Polygonzüge ermittelt werden müssen, ist die Übertragbarkeit der Aussage, daß Verfahren, die die Sichtbarkeit in der Bildebene entscheiden, den Punktetests überlegen seien, nicht von Polyedern auf räumliche Objekte mit Flächen 2. Ordnung übertragbar.

Eingabeformen

Zur Eingabe der graphischen Aufträge werden von den betrachteten Systemen Tabellen /167, 142, 164/ oder Kommandosprachen /88, 22, 20/ benutzt. Es handelt sich bei diesen Verfahren um statische Objektbeschreibungen.

Bei der Beschreibung der Objekte wird in /164/, /142/ und /22/ von einer räumlichen (materiellen) Repräsentanz ausgegangen.

Bei Becker /20/ erfolgt die Auftragserteilung an das System in einer abstrakten Kommandoschreibweise. Die Programmierung wird in expliziten Anweisungen mittels dafür vorgesehener Kürzel zum Zeichnen von einzelnen Durchdringungen (LI DU), Umrissen (LI UM) oder Endbegrenzungen (LI GR) vorgenommen.

Neben einer Kommandosprache stellt Braid /22/ für die Beschreibung der Objekte eine FORTRAN-Schnittstelle zur Verfügung. Der Anwender kommuniziert mit dem System (BUILDFORT) über Unterprogrammaufrufe.

Das Programm SPACE-PLOT /142/ erhält seinen Auftrag über eine Anzahl von Steuerkarten unterschiedlicher Kartenart. Diese Eingabe muß in einer Form codiert werden, die nur geringe Flexibilität besitzt.

Von der Möglichkeit zur Formulierung der Probleme zur Darstellung räumlicher Objekte mit einer graphischen Programmiersprache sind alle Systeme weit entfernt. Für eine sinnvolle Nutzung der Fähigkeiten solcher Programme ist jedoch die Flexibilität der Objektbeschreibung sowie die Art der Eingabe der problembestimmenden Parameter von ausschlaggebender Bedeutung.

Implementierung

Alle im Vergleich betrachteten Programmezur Behandlung von Flächen 2. Ordnung sind in FORTRAN geschrieben. Die Angaben zu den Erfordernissen der verwendeten Hardware sind nur bei wenigen Systemen enthalten. Nach den verfügbaren Unterlagen benötigen alle Programme Rechner mit einer Mindestkernspeicherausstattung von 150 K Bytes.

2.4 Datenstrukturen graphischer Systeme

Bei der Behandlung von Problemen mit dem Rechner sind die Möglichkeiten zur Abbildung eines aus der Abstraktion der zu behandelnden Aufgabe entwickelten Modells von besonderer Bedeutung. Je geringer die aus Implementierungsgesichtspunkten folgenden Einschränkungen das gewählte Modell beeinflussen, desto besser ist die vorhandene DV-Umgebung für die Realisierung der Problemlösung geeignet.

Die Eigenschaften, die ein Modell ausmachen, ergeben sich nach Ross /119/ durch das Zusammenwirken von:

- Daten,
- Struktur und
- Algorithmus.

Die Struktur eines Modells ist nach der Abbildung eines Problemes im Rechner entweder in den Daten oder in den Algorithmen repräsentiert. Der Übergang zu den anderen Komponenten des Modells ist daher fließend.

Die zur Implementierung graphischer Systeme verwendeten Programmiersprachen besitzen nur geringe Fähigkeiten zur direkten Beschreibung von komplexen Datenstrukturen. FORTRAN kennt nur statische Datenfelder, während die dynamischen Fähigkeiten von PL/1 sich auf Baumstrukturen beschränken.

Bei der Realisierung graphischer Systeme müssen mehrere Arten von Datenstrukturen unterschieden werden:

- die Struktur der graphischen Objekte selbst und
- die Struktur der Relationen unter diesen Objekten.

Unter Struktur der graphischen Objekte wird die Zusammenfassung aller geometrischen Angaben und Darstellungsattribute verstanden, die für eine graphische Ausgabe erforderlich sind.

Die Relationen zwischen Objekten beinhalten deren Verknüpfung zu komplexen aber noch rein graphischen Strukturen, wie z.B. Körpern

Diese Strukturen werden in einer realen Umgebung ergänzt durch:

- die eigentliche Problemdatenstruktur und
- die Speicherstruktur.

Die Speicherstruktur - d.h. die Abbildung der Daten im Rechner nach Hardwarebedingungen - ist hier der Vollständigkeit wegen aufgeführt, sollte jedoch klar von den übrigen unterschieden werden.

Die beiden Erstgenannten stellen nur eine ausgezeichnete Art von Problemdatenstrukturen dar. Sie sind hier deshalb von der eigentlichen Problemdatenstruktur zu trennen, da sie hier ein Hilfsmittel für die Darstellung der in diesen Daten repräsentierten Probleme der Fluiddynamik, Strukturmechanik usw. sind.

Um die oben genannten Beschränkungen der Implementierungssprachen zu beseitigen, die vor allem die Realisierung von interaktiven graphischen Systemen behindern, wurden dynamische Strukturen entwickelt. Die Arbeiten von Encarnacao /42/, Furukawa /51/, Gray /55/ und Williams /163/ vermitteln hierzu einen Überblick. Für die im graphischen Bereich erforderliche Abspeicherung sich dynamisch ändernder Abhängigkeiten haben sich neben Ringen und Bäumen vor allem assoziative Strukturen bewährt. Die Fähigkeiten zur Bildung dieser Strukturen werden in Form von Subroutinenaufrufen oder Spracherweiterungen z.B. bei APL /34/, ASP /56/ und DATAS /57/ bereitgestellt.

Diese Datenstrukturen ermöglichen eine Abspeicherung und Änderung der Relationen der graphischen Objekte in flexibeler Weise. Diese Flexibilität wird aber erkauft durch einen erheblichen Aufwand an Verarbeitungszeit und Speicherplatz, der durch die zusätzliche Abspeicherung und Interpretation von Verweisinformation verursacht wird. Bei Systemen mit großer Flexibilität übertrifft häufig der Speicheraufwand der realisierten Zeigerinformation den für die eigentlichen graphischen Attribute der Objekte.

Am Beispiel des Systems GRAPHIC /46, 47, 150, 151/, an dessen Konzipierung und Implementierung der Autor maßgeblich beteiligt war, sollen einige Konsequenzen eines überwiegend datenstrukturbezogenen Systementwurfes verdeutlich werden.

Das System GRAPHIC realisiert seine graphischen Fähigkeiten basierend auf einer assoziativen Datenstruktur als Subsystem zu ICES /120, 144/. Da die mit dem in ICES verfügbaren Subsystem CDL ('command definition language') zur Definition von Problemsprachen nur Kommandosprachen ohne Fähigkeiten einer höheren Programmiersprache verarbeitet werden können, wurden in GRAPHIC algorithmische Fähigkeiten ebenfalls in der Datenstruktur abgebildet, so daß logische Abfragen, Prozeduren, Schleifen, usw. als Objekte in der Struktur repräsentiert waren. Die Ausführung eines Programmes war dann gleichbedeutend der Interpretation und dynamischen Veränderung einer assoziativen Datenstruktur, die alle ein Modell bestimmenden Angaben zu Daten, Struktur und Algorithmus enthielt (vgl. 3.2). Diese Verarbeitungsweise erfordert jedoch einen großen Rechenaufwand.

Die bei der Erstellung und Anwendung des in GRAPHIC verwirklichten Konzepts gemachten Erfahrungen lieferten wichtige Erkenntnisse für die hier beschriebene alternative Realisierung, die eine algorithmische Behandlung der graphischen Aufgaben anstrebt.

2.5 Folgende Entwurfsziele für ein neues Konzept

Aus den obigen Vergleichen lassen sich folgende Tatsachen festhalten:

Es gibt keine Sprache, die

- bei Verfügbarkeit der Fähigkeiten einer höheren Programmiersprache einen ausreichenden Vorrat an graphischen Objekttypen besitzt,
- die Formulierung von graphischen Problemen durch einen ausreichenden Vorrat an graphischen Operationen für diese Objekte bereitstellt,
- innerhalb eines integrierten Systems verfügbar ist, und daher mit anderen Problembereichen Daten austauschen kann,
- Körper, die durch eine Auswahl von Flächen 2. Ordnung umschlossen sind, kennt,
- die Beschreibung dieser Körper algorithmisch vornimmt und die
- die Problemdaten in Koordinatenachsen behandelt.

Es gibt kein System, das

- die Probleme der Liniendarstellung von Objekten mit Flächen 2. Ordnung analytisch angeht,
- sich bei der internen Darstellung auf das Minimum an geometrisch notwendiger Information beschränkt,
- die algorithmische Beschreibung von Körpern mit Flächen 2. Ordnung erlaubt und das
- beim Sichtbarkeitsalgorithmus ebenfalls die analytischen Ansätze benutzt.

Die Erfahrung in der Programmierung graphischer Probleme zeigt jedoch, daß alle in der obigen Zusammenstellung als bisher nicht verfügbar festgehaltenen Eigenschaften bei graphischen Anwendungen unerläßlich sind.

Das hier vorgestellte System soll deshalb für die graphische Programmierung eine Sprache mit einem der Problematik der Behandlung technischer Abbildungen angemessenen Objekt- und Operationsvorrat unter Beibehaltung der Fähigkeiten höherer Programmiersprachen bereitstellen. Durch die Einbettung in ein integriertes System sind diese Fähigkeiten auch aus anderen Problembereichen zur Darstellung der Zusammenhänge nutzbar.

Die Körperbehandlung soll durch effektive Methoden zur Beschreibung und Darstellung der Anwendung im konstruktiven Bereich zugänglich gemacht werden. Ein Flächenkatalog aus Ebene, Kugel, Zylinder und Kegel soll die Beschreibung komplizierter Körper ermöglichen.

Die in der Praxis geforderte Flexibilität wird durch eine algorithmische Beschreibung der Objekte unter Nutzung von Lineartransformationen und den Möglichkeiten zur beliebigen Projektion angestrebt.

Die Darstellung soll unter Anwendung eines Visibilitätsverfahrens erfolgen.

3. Realisierung und Eigenschaften der GIPSY-Sprache

Da die Realisierung von GIPSY als Subsystem zum integrierten System für den rechnergestützten Entwurf REGENT /48, 45/ vorgenommen wurde, ist für das Verständnis der folgenden Abschnitte ein kurzer Überblick über die Fähigkeiten integrierter Systeme und insbesondere über REGENT und sein Sprachdefinitionssystem PLS /45/ erforderlich.

3.1 Definierte Sprachen integrierter Systeme

Im Bereich des rechnergestützten Entwurfs (CAD)wird die Notwendigkeit von integrierten Systemen zunehmend erkannt und deren Nützlichkeit gewürdigt /27, 119, 143, 144, 147/. Neben einer Vielzahl von Programmsystemen, die die koordinierte Ausführung einer großen Anzahl von Routinen zur Lösung eines streng umrissenen Problemkreises - wie z.B. Reaktorphysik oder -dynamik - zulassen, sind hier diejenigen Entwicklungen zu nennen, die die Entwicklung von Subsystemen für verschiedenartigste Anwendungen unterstützen und deren Anwendung erleichtern. Es sind dies:

- ICES /120/,
- AED /118/,
- PLAN /74/,
- POLO /94/,
- GENESYS /6/,
- IST /117/ und
- REGENT /48/.

Seit ICES (Integrated Civil Engineering System) ist allen Systemen die Gliederung in

- Systemkern und
- Subsysteme

gemein, wobei im Systemkern diejenigen Fähigkeiten zusammengefaßt sind, um die die Basissprache - außer bei AED und REGENT ist dies FORTRAN - für die Systemprogrammierung erweitert wurde, wie z.B. :

- dynamisches Aufrufen von Moduln ('dynamic linking'),
- dynamische Anpassung von Datenfeldern an die Problemanforderungen ('dynamic arrays'),
- Auslagerung dieser Daten temporär oder permanent auf externe Datenträger ('virtual memory', 'database'),
- Definition und Übersetzung von Problemsprachen und
- Bereitstellung von Testhilfen.

Für die weiteren Betrachtungen ist vor allem die Möglichkeit zur Definition und Übersetzung von Problemsprachen von besonderem Interesse. Bestreben jeder Definition einer problemorientierten Sprache (Problem Oriented Language (POL)) ist das Ziel /146/:

Modellbeschreibung = maschinenlesbare Darstellung.

Es ergibt sich dann die Möglichkeit, die für die Kommunikation zwischen Fachleuten in einem Problembereich etablierte Terminologie als rechnerverständliche Eingabe zu verwenden, wobei daraus

- ein geringer Lernaufwand für den Anwender,
- eine kleinere Fehlerwahrscheinlichkeit und
- eine weitgehend selbstdokumentierende Eingabe

folgt.

Neben der Formulierung mathematischer Probleme durch ALGOL, FORTRAN, PL/1 oder APL/53/ sind inzwischen auch Sprachen für

- Textverarbeitung (SNOBOL),
- Simulationen (SIMULA, CSMP),
- Schaltungsanalyse (ECAP),
- Werkzeugmaschinensteuerung (EXAPT) oder
- Teilgebiete der Mathematik, wie Lösung partieller Differetialgleichungen (PDEL)

bekannt.

Einen Überblick über die heutige Sprachenvielfalt und einen Ausblick auf erwartete Entwicklungen geben Rosen /121/ und Sammet /139/. Mit spezialisierten Sprachen integrierter Systeme befassen sich wiederum Sammet /134/ und Schlechtendahl /146/. Die definierten Sprachen von ICES, PLAN, POLO und IST sind reine Kommandosprachen, während GENESYS auch auf der POL-Ebene FORTRAN unterstützt. Graphische Fähigkeiten sind bei IST als Bestandteil des Kernes /140, 64/ bei GENESYS als Subsystem (GINO) /65/ verfügbar. Aus dem

weiten Spektrum der unter ICES realisierten Sprachen seien nur CDL, PROJEKT, TOPOLOGY, COGO, STRUDL, GETAM und GRAPHIC /12, 32, 36, 47, 49/ genannt.

Das bereits weiter oben erwähnte Subsystem GRAPHIC erlaubte unter ICES die Behandlung der Probleme zweidimensionaler Strichgraphik. Die Definition der Sprache erfolgte mittels CDL, die eigentlich nur die Vereinbarung von Kommandosprachen (vergl. Abb. 2.3) erlaubt. Die Erweiterung der Spracheigenschaften um die Fähigkeiten höherer Programmiersprachen wie Prozedur- oder Schleifenvereinbarung und Kontroll- oder Arithmetik-Anweisungen konnte nur durch die gemeinsame Abspeicherung aller Operationen (Algorithmen) mit den Objekten in einer komplexen Datenstruktur gelöst werden (vgl. Abschnitt 2.4).

Die Erfahrungen bei der Anwendung dieser problemangepaßten Wortsprache haben gezeigt, daß neben der Problemorientiertheit folgende wesentliche Merkmale einer höheren Programmiersprache nicht fehlen dürfen:

- Neben Konstanten muß die Angabe von Ausdrücken - arithmetischer oder logischer Art - unter Verwendung von namentlich identifizierten Variablen und Datenaggregaten möglich sein, um die Ergebnisse von Zwischenrechnungen oder die Variation von Einflußgrößen zu Parameterstudien des gleichen Modells nutzbar zu machen.
- Der Ablauf des POL-Programmes muß steuerbar sein, d.h. Schleifen und bedingte Anweisungen sollten eine algorithmische Beschreibung des Problems erlauben.
- Für die wiederholte Ausführung von Problemen bekannter Struktur sollte die Zusammenfassung von Statements zu Unterprogrammen und deren Abspeicherung - wenn möglich in übersetzter Form - möglich sein.
- Möglichkeiten zum Einlesen modellbeschreibender Daten sollten verfügbar sein.

Das Fehlen dieser Eigenschaften und andere Implementierungsmerkmale führten zur Realisierung eines alternativen Konzepts mit REGENT /48, 145/. Es soll hier nur soweit auf REGENT eingegangen werden, wie dies zum Verständnis der Umgebung von GIPSY - als einem REGENT-Subsystem - nötig ist.

3.2 Eingliederung in REGENT

Im REGENT-System wird die Definition und Übersetzung von Problemsprachen durch das von Enderle entwickelte Subsystem PLS (Problem Language System) /45/ übernommen, das seinerseits ebenfalls ein Subsystem darstellt. Zu den wichtigsten Eigenschaften der mit PLS definierten POLs zählen:

- das Enthaltensein des vollen PL/1-Sprachumfangs und damit gute Möglichkeiten zur Strukturierung der Programme und zur Kontrolle des Programmablaufes,
- eine einheitliche Sprachbasis für alle Subsystemsprachen, wodurch bei Einhaltung der Blockstruktur eine Möglichkeit zum Datenaustausch zwischen Subsystemen gegeben ist und
- die Anwendung einer Vorübersetzungstechnik bei der Abarbeitung der Problemstatements.

Die Ausführung der Definition einer Problemsprache mit PLS sowie deren Anwendung wird in Abb. 3.1 deutlich.

Die Subsysteme des REGENT-Systems gliedern sich in

- Problemsprache,
- Datenstruktur,
- Module (semantische Routinen),
- Nachrichten und
- Datenbasis,

die eine Entsprechung in den Komponenten des Systemkerns finden.

Auf diese Fähigkeiten aufbauend wurde die graphische Problemsprache entwikkelt, die die Behandlung graphischer Aufgaben erlaubt, ohne die Eigenschaften höherer Programmiersprachen vermissen zu lassen.

Für die Anwendung von problemorientierten Sprachen durch DV-Unkundige wird deren leichte Erlernbarkeit und Anwendbarkeit gefordert. Dies setzt voraus, daß es dem Anwendungsprogrammierer ermöglicht ist, den in dem zu behandelnden Fachbereich üblichen Sprachgebrauch zur Formulierung seines Problembereichs zu nutzen. Ein Konstrukteur möchte die ihm vertrauten Objekte in der Sprache, die ihm zur Beschreibung einer Aufgabe zur Verfügung steht, wiederfinden. D.h. im graphischen Bereich müssen die Objekte - wie Punkt, Linie, Text oder auch Ebene, Kugel, Zylinder usw. -, in denen ein Anwender denkt, wenn er sich mit dem Problem auseinandersetzt, in der Sprache enthalten und

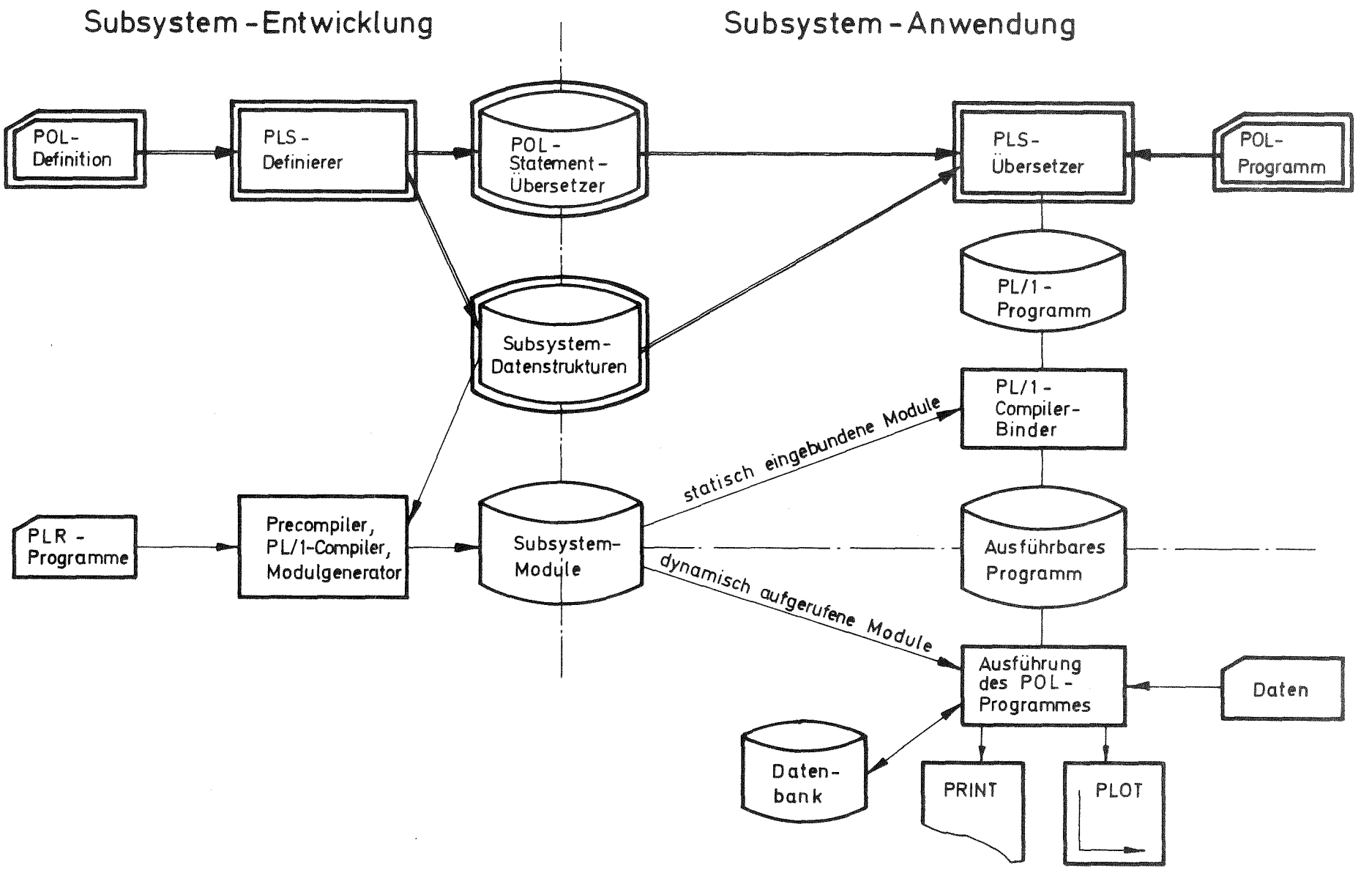


Abb. 3.1: Anwendung des REGENT-Systems nach Enderle /45/.

einfach manipulierbar sein. Er kann diese Objekte, wenn seine mangelnden DV-Fähigkeiten dies erzwingen, in einfachen, auf Kommandoform reduzierten Folgen von Statements behandeln, ohne den vollen Sprachumfang kennen zu müssen. Für den geübten Programmierer bietet das Vorhandensein der Eigenschaften einer höheren Programmiersprache jedoch erhebliche Vorteile, um auch komplizierte Aufgaben mit Hilfe des Systems zu lösen, zumal die Regeln für die Anwendung der graphischen Erweiterung aus der von der Basisprache vertrauten Syntax folgen.

Das Ablaufschema einer GIPSY-Anwendung wird in Abb. 3.2 verdeutlicht. Das um die graphischen Anweisungen erweiterte Programm wird eingelesen und vom mit PLS definierten GIPSY-Übersetzer mittels abgespeicherter Treiberroutinen und Datenstrukturen in ein gültiges Programm der Gastgebersprache PL/1 übertragen. Die anschließende Erzeugung eines ausführbaren Programms erfolgt nach den üblichen Regeln eines Übersetzungs- und Bindelaufes (Compile und Link). Beim Binden wird ein Teil der semantischen Routinen der graphischen Spracherweiterung statisch in das Programm eingebaut. Es handelt sich dabei um diejenigen Routinen, die wegen ihrer hohen Ansprechfrequenz oder wegen DV-technischer Implementierungsgesichtpunkte sinnvollerweise fest eingebunden werden. Der größere Anteil der semantischen Routinen wird nur im Bedarfsfalle in den Kernspeicher der Anlage geholt. Diese durch die Modulverwaltung des REGENT-Systemkerns unterstützte Technik kann zu einer erheblichen Reduzierung des Speicherplatzbedarfes führen.

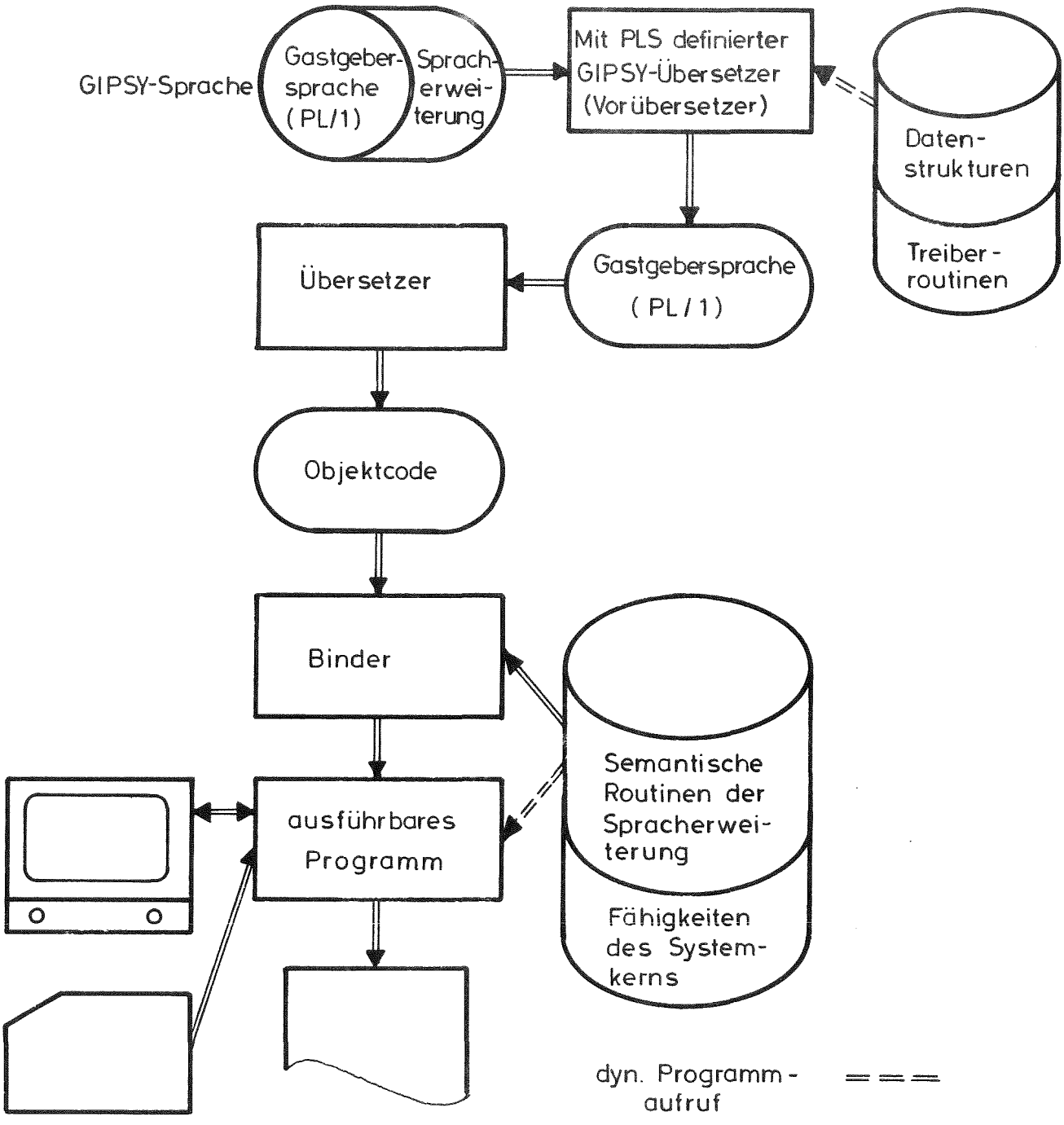


Abb. 3.2: Ablauf der Ausführung eines GIPSY-Programmes

3.3 Arten der GIPSY-Statements

In GIPSY wird versucht, die graphischen Fähigkeiten möglichst nahtlos in die von PL/1 vorgeprägte Umgebung einzupassen. In Anlehnung an die PL/1 Sprachbeschreibung /73/ ergibt sich daraus die in Abb. 3.3 dargestellte Eingliederung der Erweiterung in die PL/1-Umgebung. Eine vollständige Liste der Statements der GIPSY-Sprache kann mittels PLS erstellt werden. Die Liste in Abb. 3.4 enthält eine Kennzeichnung der Anweisungen nach

- PL/1 -
- POL - und
- Systemstatements.

In den folgenden Abschnitten soll auf die graphischen Komponenten der GIPSY-Sprache näher eingegangen werden.

3.3.1 Graphische Datentypen

Die in Abschnitt 2 untersuchten graphischen Systeme waren hinsichtlich ihres Objektumfangs unzureichend für die Behandlung graphischer Aufgaben ausgestattet. In GIPSY wird das Ziel angestrebt, sowohl für die zwei- und dreidimensionale Strichgraphik als auch die Beschreibung von Körpern einen ausreichenden Katalog von geeigneten graphischen Objekten anzubieten, die zudem durch eine Vielzahl von graphischen Operationen erzeugt oder manipuliert werden können. Für die Formulierung der graphischen Probleme von Strichgebilden (Eigenausdehnung der Objekte <2) werden Problemdaten von PL/1 um folgende graphische Daten ergänzt:

- Punkte (POINT),
- Texte (TEXT),
- Polygone (POLYGON),
- Kreise (CIRCLE),
- Kreisbögen (ARC),
- Koordinatenachsen (AXIS),
- Kurven mit Problemwerten in diesen Koordinatenachsen (CURVE) und
- Kollektion der bisher genannten Objekte (COLLECTION).

PL/1	GIPSY
Data Types:	
*Problem Data: arithmetic, string, graphical arithmetic string graphical : POINT, TEXT, POLYGON, CIRCLE, ARC, AXIS, CURVE, COLLECTION, PLANE, BALL, CYLINDER, CONE, SPACE, COLLECTION *Program Control Data: LABEL,...	
Statements:	
* Compound Statements: IF, ON * Simple Statements: keyword, assign, null	
Classes	
- Descriptive: DCL, DEFAULT (,OPEN), .. - Data Movement and Computational: assignment graphical assignment FILL - I/O: record: READ, ... stream: GET, PUT, STATUS, PRINT control: OPEN, CLOSE, ... ,CHANGE, EDIT graphical: PLOT - Storage Control: ALLOCATE, FREE - Control: GOTO, IF, DO, CALL, RETURN, ... - Exeption Control: ON, ... - Preprocessor: % DCL, ... - Program Organisation: PROC, ENTRY, ... , END, DEFINE - Diagnostic: SNAP, FLOW, DATA, ...	

Abb. 3.3: Einordnung der graphischen Spracherweiterung in das PL/1-Konzept

```
**********************************
*                                *
*      SUBSYSTEM - INFORMATION   *
*                                *
**********************************
```

NAME OF SUBSYSTEM	PREFIX	CREATION DATE	SUB-SYSTEM LIST	ABBREV. ?	NUMBER OF STATEMENTS	NUMBER OF CLASSES	NUMBER OF DATA-DECL.
GIPSY	GI	11.07.75	PLGIPS	NO	79	5	2

NAME OF STATEMENT	CHARACTERISTIC	MODULL
%AS	PL/1-STATEMENT	
ASS	PL/1-STATEMENT	
%ACT	PL/1-STATEMENT,ALIAS	
%ACTIVATE	PL/1-STATEMENT	
%CONTROL	PL/1-STATEMENT	
%DCL	PL/1-STATEMENT,ALIAS	
%DEACT	PL/1-STATEMENT,ALIAS	
%DEACTIVATE	PL/1-STATEMENT	
%DECLARE	PL/1-STATEMENT	
%DO	PL/1-STATEMENT	
%END	PL/1-STATEMENT	
%GO	PL/1-STATEMENT	
%GOTO	PL/1-STATEMENT,ALIAS	
%IF	PL/1-STATEMENT	
%INCLUDE	PL/1-STATEMENT	
%PAGE	PL/1-STATEMENT	
%SKIP	PL/1-STATEMENT	
ALLOC	PL/1-STATEMENT,ALIAS	
ALLOCATE	PL/1-STATEMENT	
BEGIN	PL/1-STATEMENT	
CALL	PL/1-STATEMENT	
CHANGE	POL-STATEMENT	GICHAN
CHECK	PL/1-STATEMENT	
CLOSE	PL/1-STATEMENT	
DCL	POL-STATEMENT,ALIAS	GIDECL
	SUBST. FOR PL/1-STATEMENT,ALIAS	
DECLARE	POL-STATEMENT	GIDECL
	SUBST. FOR PL/1-STATEMENT	
DEFAULT	PL/1-STATEMENT	
DEFINE	POL-STATEMENT,ABBREV.	GIDEFI
DELAY	PL/1-STATEMENT	
DELETE	PL/1-STATEMENT	
DFT	PL/1-STATEMENT,ALIAS	
DISPLAY	PL/1-STATEMENT	
DO	PL/1-STATEMENT	
EDIT	POL-STATEMENT	GIEDIT
ELSE	PL/1-STATEMENT	
END	POL-STATEMENT	GIENDE
	SUBST. FOR SYSTEM STATEMENT	
ENTER	SYSTEM STATEMENT,ABBREV.	GCENTE
ENTRY	PL/1-STATEMENT	
EXIT	PL/1-STATEMENT	

NAME OF STATEMENT	CHARACTERISTIC	MODUL
FETCH	PL/1-STATEMENT	
FILL	POL-STATEMENT	GIFILL
FINISH	SYSTEM STATEMENT,ABBREV.	GCFINI
FLOW	PL/1-STATEMENT	
FORMAT	PL/1-STATEMENT	
FREE	PL/1-STATEMENT	
GET	PL/1-STATEMENT	
GO	PL/1-STATEMENT	
GOTO	PL/1-STATEMENT,ALIAS	
IF	PL/1-STATEMENT	
LOCATE	PL/1-STATEMENT	
NOCHECK	PL/1-STATEMENT	
NOFLOW	PL/1-STATEMENT	
ON	PL/1-STATEMENT	
OPEN	POL-STATEMENT	GIOPEN
	SUBST. FOR PL/1-STATEMENT	
PLCT	POL-STATEMENT	GIPLCT
PCCLSIZE	SYSTEM STATEMENT,ABBREV.	QQPCOL
PRINT	POL-STATEMENT	GIPRIC
	SUBST. FOR SYSTEM STATEMENT	
PROC	PL/1-STATEMENT,ALIAS	
PROCEDURE	PL/1-STATEMENT	
PUT	PL/1-STATEMENT	
READ	PL/1-STATEMENT	
RELEASE	PL/1-STATEMENT	
RETURN	PL/1-STATEMENT	
REVERT	PL/1-STATEMENT	
REWRITE	PL/1-STATEMENT	
SET	POL-STATEMENT	GISET
SIGNAL	PL/1-STATEMENT	
STATUS	POL-STATEMENT	GISTAT
STOP	PL/1-STATEMENT	
SYNTAX	PCL-STATEMENT,ABBREV.	GISYNT
TEST	POL-STATEMENT	GITEST
TRACE	SYSTEM STATEMENT,ABBREV.	QQTRAC
UNLOCK	PL/1-STATEMENT	
WAIT	PL/1-STATEMENT	
WRITE	PL/1-STATEMENT	
GR_CCRAN	CLAUSE	GIGRCO
GR_ELEMENT	CLAUSE	GIGREL
GR_ERROR	CLAUSE	GIGRER
GR_NAME	CLAUSE	GIGRNA
GR_PLPRT	CLAUSE	GIGRPL
CCMMCN	DATA-DECLARATION	GICOMI
GRDCL	DATA-DECLARATION	GIGRDC

Abb. 3.4: Liste der in GIPSY verfügbaren Statements (erstellt mit PLS/45/).

Die ersten Objekte dieser Auswahl - Punkte, Texte, Polygone und Kreise - werden in zwei und drei Dimensionen unterstützt.
Zum Aufbau räumlicher Strukturen (Eigenausdehnung ≥2) können die Flächen:

- Ebene (PIANE),
- Kugel (BALL),
- Zylinder (CYLINDER) und
- Kegel (CONE)

herangezogen werden, aus denen nach weiter unten noch zu erläuternden Regeln

- Raumelemente (SPACE) und
- Körper (COLLECTIONEN)

aufgebaut werden können.

Keines der aus der Literatur bekannten und in Kapitel 2 untersuchten Systeme bietet einen vergleichbaren Satz von graphischen Objekten an. Es findet sich in der Literatur nur ein System, das in ähnlicher Weise die Speicherung und Verarbeitung graphischer Information anstrebte. Die von Comba /28/ vorgeschlagene 'Geometry Language (GL)' geht von einer ähnlichen Objektauswahl aus. Comba weist aber ausdrücklich darauf hin, daß keine Implementierung des Konzeptes vorgenommen wurde. Seine Arbeit befaßt sich mehr mit der Notwendigkeit für ein solches Konzept und mit den Wünschen zu seiner Realisierung.

Die Bereitstellung eines großen Kataloges an zur Beschreibung des graphischen Problemes nutzbaren Objekten ist ein Aspekt der graphischen Sprache, die Möglichkeit zur Behandlung dieser Objekte mit einer für arithmetische Anwendungen bewährten algorithmischen Form ist eine ebenso wichtige Aufgabe. Deshalb werden über die Fähigkeiten von Kommandosprachen hinausgehende Forderungen nach der Verwendung von

- Variablennamen,
- arithmetischen Ausdrücken anstelle von Konstanten und
- Anweisungen zur Programmablaufsteuerung

gestellt. Durch die Heranziehung von PLS zur Sprachdefinition steht für die Problemsprache des Subsystems - sofern der Subsystemprogrammierer dies nicht absichtlich unterbindet - der volle PL/1-Sprachumfang mit den Möglichkeiten zur:

- Vereinbarung von Prozeduren,
- Schachtelung von Blöcken,
- Verwendung von Schleifen,
- Kontrolle des Programmablaufes,
- Deklaration und Übergabe von Datenaggregaten,
- Datenein- und -ausgabe, usw.

zur Verfügung.

In GIPSY mußte deshalb die Implementierung der graphischen Objekte so vorgenommen werden, daß die Nutzung der obengenannten PL/1-Fähigkeiten nicht nur nicht verhindert wurde, sondern auch bei der graphischen Erweiterung anwendbar blieben.

3.3.1.1 Möglichkeiten der Objektimplementierung

Zu den grundlegenden Eigenschaften einer Programmiersprache gehört die Fähigkeit, die Objekte der Sprache durch Namen zu identifizieren. Für die graphischen Objekte ist eine Realisierung anzustreben, die diese Identifikationsart nicht nur bei Skalaren, sondern auch bei Aggregaten, Argumenten, usw. erlaubt, um so die graphische Aufgabe unter Nutzung dieses erweiterten Sprachumfangs programmierbar zu machen. Für die gemeinsame Speicherung von Werten unterschiedlichen Datentypes sind in PL/1 Strukturen geeignet. Sie sollen zur Objektrepräsentation herangezogen werden, da die zu handhabenden graphischen Objekte neben den geometrischen Attributen auch Angaben über ihre Darstellung enthalten. Für die namentliche Identifikation dieser Strukturen stehen prinzipiell drei Wege offen

- Name der Struktur
- Name eines auf die Struktur weisenden Pointers und
- die Selbstreferenztechnik.

Die Vor- und Nachteile dieser Methoden sind in Abb. 3.5 gegenübergestellt, wobei als Entscheidungskriterien für Auswahl die Möglichkeit der Realisation von

- Deklarationen,
- Operationen und
- Zuweisungen

gelten. Nach diesem Vergleich kommt zur Darstellung der graphischen Objekte nur die für GIPSY entwickelte Technik des Selbstreferierens in Frage, da sie als einzige die Implementation aller gewünschten Eigenschaften ermöglicht.

Identifikation durch / Kriterium der Beurteilung		Namen der Struktur (1)	Namen eines auf eine Struktur weisenden Pointers (2)	Selbstreferenz (3)
Deklaration	POL	DCL A POINT;	DCL A GRAPH_ELEM ;	DCL A POINT;
	resultierend	DCL 1 A, 2 TYP, 2 ATTRIBUTE;	DCL A POINTER; DCL 1 GR_E_A BASED(A), 2 TYP, 2 ATTRIBUTE;	DCL 1 POINT_A, 2 A POINTER INIT(ADDR(A)), 2 TYP, 2 ATTRIBUTE;
Operation	POL	POINT(, ,)	POINT(, ,)	POINT(, ,)
	resultierend	POINT:PROC(, ,)RETURNS(*); RETURN(POINT_STRUKT);	POINT:PROC(, ,)RETURNS(PTR); ALLOCATE POINT_STRUKT SET(PS); RETURN(PS);	POINT:PROC(, ,)RETURNS(PTR); ALLOCATE POINT_STRUKT SET(PS); RETURN(PS);
Zuweisung	POL	A=B, A=POINT(, ,);	SET A=B; SET A=POINT(, ,);	SET A=B; SET A=POINT(, ,);
	resultierend	identisch (Zuweisung wird von PL/1 gemacht, PL/1 zerstört das Funktionsergebnis selbst.)	CALL ASSIGN(A,B); CALL ASSIGN(A,POINT(, ,)); (Zuweisung und Zerstörung muß programmiert werden.)	CALL ASSIGN(A,B); CALL ASSIGN(A,POINT(, ,)) (Zuweisung und Zerstörung muß programmiert werden.)
Bemerkungen		Diese Implementierung würde die einfachste und effektivste Realisierung darstellen, wenn eine PL/1-Funktionsprozedur Strukturen zurückliefern könnte.	Die graph.Elemente und damit die POINTER sollten beliebige 'storage class' besitzen. Da die Strukturen aber BASED sind, besteht die Gefahr, daß Strukturen liegen bleiben, deren Pointer inzwischen zerstört sind (und umgekehrt).	Bietet die Vorteile der Methode (1) hinsichtlich Deklaration - allerdings mit Initialisierungsaufwand für Selbstreferenz - und macht Zuweisung und Operationen nach Methode (2) möglich.

Abb. 3.5: Möglichkeiten der Implementierung graphischer Daten in PL/1

3.3.1.2 Realisierung der Deklarationen von graphischen Objekten

Nach dem oben Dargelegten wurden die graphischen Objekte durch eine Selbstreferenz identifiziert, die für jedes graphische Element einen Verweis auf sich selbst bedeutet. Diese Referenz wird bei der Allokierung des Objektes durch eine spezielle Routine hergestellt, deren Aufruf bei der Umsetzung der Deklarationen mitgeneriert wird. Zur Erklärung der im folgenden verwendeten Syntaxbeschreibung sei auf Anhang A verwiesen. Die Deklaration für graphische Daten hat folgendes Aussehen:

DCL <ident> [<dim>]<type>[<length>][<storage class>];

Mit dieser Anweisung werden, wie von der Deklaration normaler PL/1-Variablen vertraut, graphische Problemvariable vereinbart, deren Namen (<ident>) nach normalen PL/1-Regeln aufgebaut ist. Der Attributsatz für diese Deklaration ist aus

- der Dimensionsvereinbarung (<dim>),
- dem Datentyp (<type>),
- der Längenangabe (<length>) für Texte, Polygonzüge, Achsen und Raumelemente sowie
- der Speicherklasse (<storage class>)

aufgebaut.

Die Bedeutung und die Möglichkeiten dieser Attribute sollen durch die Statements des Beispielprogrammes in Abb. 3.6 verdeutlicht werden, das keine sinnvolle graphische Aufgabe beinhaltet, sondern nur der Demonstration der PL/1-artigen Behandlung der graphischen Daten dient. In den Anweisungen, die durch die Angabe einer Nummer in den Klammern gekennzeichnet sind, werden folgende Eigenschaften ersichtlich:

- Vereinbarung von Feldern mit beliebiger Dimension und Feldgrenzen (4),
- Anlegen von Automatic-Variablen mit problemabhängigen Längen (7) unter Beachtung der Blockstruktur,
- Zusammenfassung von graphischen Daten unterschiedlichen Types in Strukturen (11), wobei Strukturen von Feldern erlaubt sind, sowie

```
1       SAMPLE:PROC OPTIONS(MAIN) REGENT (NOCOA);
2       DCL VALUE DEC FLOAT(6) INIT (0.01);
        /*****************************************************************/
3       ENTER GIPSY;
4           DCL PUNKT(-5:5) POINT;
5           DCL NP BIN FIXED(15) INIT (3);
6           BEGIN;
7             DCL STRICH(NP+1) POLY(NP);
8               SET STRICH(1)=POLY(CCLL(PUNKT(-5)+PUNKT(0)+PUNKT(5)));
9               PRINT(SHIFT(STRICH(1),10 CM,2.,NP MM));
10          END;
11          DCL 1 OBJECT,
12                2 P(2) POINT,
13                2 SUBOBJECT,
14                  3 A(3) CIRCLE,
15                  3 B POINT;
16          DO I=1 TO 2;
17            SET OBJECT.P(I)=ROT(SHI(PUNKT(I),VALUE,COS(VALUE),0.),45.,0.,0.);
18          END;
19          SET OBJECT.SUBOBJECT.A(2)=CIRCLE(P(1),OBJECT.SUBOBJECT.B,VALUE);
20          SET OBJECT.P(1)=POINT(10.,10.**2,10.*10./5. CM);
21          POINT_PUNKT(0).X(*)=5.*VALUE;
22          CALL SUB(PUNKT(0),OBJECT.P);
        /*.............................................................*/
23      SUB:PROC(PS,PA);
24            DCL P_PB POINTER;
25            DCL (PS,PA(2)) POINT PARAMETER,
26                PB POINT CONTROLLED,
27                PC POINT BASED (P_PC),
28                T3 TEXT(15),
29                DREI POLY(4);
30              SET DREI = POLY(CCLL(PS+PA(1)+PA(2)+PS));
31              DEF P1=NPOINT(2,DREI);
32              ALLOCATE POINT_PB,POINT_PC;
33              SET PB=PA(2);
34              SET PC=PS;
35              SET T3=FUN(PA,PC,' AUS FUN ');
36              PRINT(DREI,P1,OBJECT.SUBOBJECT.A(2),T3,PS,PB,PC);
        /*.............................................................*/
37      FUN:PROC(PF,PE,T) RETURNS(POINTER);
38            DCL (PF(2),PE) POINT PARAMETER,
39                STR TEXT(10),
40                T CHAR(*);
41              SET PE=SHIFT(PS,15 CM,0.,0.3 INCH);
42              SET STR=TEXT('ZURUECK'||T,PF(1),PE);
43              PRINT(PE,STR,PF(1),PF(2));
44              RETURN(STR);
45      END FUN;
        /*.............................................................*/
46      END SUB;
        /*.............................................................*/
47      END GIPSY;
        /*****************************************************************/
48      FINISH;
49      END SAMPLE;
```

Abb. 3.6: Beispielprogramm zur Demonstration der Objekthandhabung in GIPSY

- Verwendung der Speicherklassen CONTROLLED (26),
  BASED (27) und PARAMETER (25).

Die Speicherklasse PARAMETER muß in der Basissprache PL/1 nicht explizit deklariert werden, da dort der Compiler aufgrund der Gültigkeitsregeln für Variablennamen die Zuordnung zwischen den Namen in der Parameterliste und den Variablen des Programmes herstellt. In PLS, das, als Einpaß-Compiler realisiert, keine Gedächtnisfunktion über Statementgrenzen besitzt, ist eine Auflösung der die Gültigkeit von Namen beeinflussenden Blockstruktur eines Programmes nur mit großem zusätzlichem Aufwand möglich, und unterbleibt daher. Die Vereinbarung dieses Attributes wird dem Anwender überlassen, der sich bei richtiger Anwendung dieses Unterschiedes bewußt sein muß.

Allgemein gilt jedoch, daß eine derartige Flexibilität der GIPSY-Sprache nicht erreichbar gewesen wäre, wenn im REGENT-System nicht eine Möglichkeit zur Definition und Verarbeitung von Problemsprachen mit PLS bestanden hätte. Eine frühere Version von GIPSY, die nur die Fähigkeiten des PL/1-Preprocessors ausnutzte, führte trotz höheren Aufwandes bei der Programmierung zu weniger befriedigenden Ergebnissen hinsichtlich verarbeitbarer Syntax und Übersetzungszeit.

Die Expansion der Deklarationen, die nach folgender Vorschrift (außer bei Parametern) abläuft

```
DCL  1  POINT_<ident>[<dim>][<storage class>],
        2 <ident> POINTER,
        2 TYPE BIN FIXED(15),
        2 individueller Text (type) INIT
        CALL GROBINI(ADDR(POINT_<ident>,ntype,f(dim),1);
```

wird unter Zuordnung der obengenannten Statementnummern aus Abb. 3.7 ersichtlich.

3.3.2 Zuweisung graphischer Daten

Eine Programmierung mit graphischen Variablen erfordert eine Möglichkeit zur Übertragung, Manipulation und Abspeicherung der graphischen Informationen. Zu diesem Zweck wurde in GIPSY ein graphisches Assignment-Statement eingeführt, das den Fähigkeiten einer normalen Zuweisung z.B. im arithmeti-

SAMPLE

```
/*    4*/
    DECLARE   1 POINT_PUNKT ( -5 : 5 )  ,
          2 PUNKT PTR,
          2 TYPE BIN FIXED(15),
          2 SYMBOL BIN FIXED(15),
          2 P_UP PTR,
          2 HEIGHT DEC FLOAT(6),
          2 X(3) DEC FLOAT(6) INIT
          CALL GRCBINI(QQ,ADDR( POINT_PUNKT ),   1 , (5--5+1) ,1) ;

/*    7*/
      DECLARE   1 POLYGON_STRICH ( NP +1 )  ,
            2 STRICH PTR,
            2 TYPE BIN FIXED(15),
            2 SYMBOL BIN FIXED(15),
            2 P_UP PTR,
              :
            2 N_MAX BIN FIXED(15)INIT( ( NP +1 ) NP ),
            2 X( NP ,3)DEC FLOAT(6) INIT
            CALL GROBINI(QQ,ADDR( POLYGON_STRICH ),   3 , NP +1 ,1) ;

/*    8*/
        CALL GRASIGN ( QQ , STRICH(1) ,
        QQ_GRPOLY_E(QQ_GRPOLY_P,QQ_GRCOLO6_E(QQ_GRCOLO6_P, 3,PUNKT(-5),
PUNKT(0),PUNKT(5),NULL(),NULL(),NULL())
        )
        ,'1'B);

/*   11*/
    DECLARE       1 OBJECT ,
/*   12*/   2 POINT_P ( 2 )  ,
               3 P PTR,
               3 TYPE BIN FIXED(15),
               3 SYMBOL BIN FIXED(15),
               3 P_UP PTR,
               3 HEIGHT DEC FLOAT(6),
               3 X(3) DEC FLOAT(6) INIT
               CALL GRCBINI(QQ,ADDR( POINT_P ),   1 , 2 ,1) ,
/*   13*/   2 SUBOBJECT ,
/*   14*/     3 CIRCLE_A ( 3 )  ,
                 4 A PTR,
                 4 TYPE BIN FIXED(15),
                 4 PADDING BIT(16),
                 4 P_UP PTR,
                 4 M(3)DEC FLOAT(6),
                 4 N(3)DEC FLOAT(6),
                 4 R DEC FLOAT(6) INIT
                 CALL GRCBINI(QQ,ADDR( CIRCLE_A ),   4 , 3 ,1) ,
/*   15*/     3 POINT_B ,
                 4 B PTR,
                 4 TYPE BIN FIXED(15),
                 4 SYMBOL BIN FIXED(15),
                 4 P_UP PTR,
                 4 HEIGHT DEC FLOAT(6),
                 4 X(3) DEC FLOAT(6) INIT
                 CALL GRCBINI(QQ,ADDR( POINT_B ),   1 , 1 ,1) ;

/*   17*/
       CALL GRASIGN ( QQ , OBJECT.P(I) ,
       QQ_GRLTRAN_E(QQ_GRLTRAN_P,QQ_GRLTRAN_E(QQ_GRLTRAN_P,PUNKT(I),2,VA
LUE,COS(VALUE),0.E0), 3,45.E0,0.E0,0.E0)
       ,'1'B);
```

SUB

```
/*   25*/
      DECLARE PS PTR,
        1 POINT_PS BASED( PS ) ,
          2 HANGER PTR,
          2 TYPE BIN FIXED(15),
          2 SYMBOL BIN FIXED(15),
          2 P_UP PTR,
          2 HEIGHT DEC FLOAT(6),
          2 X(3) DEC FLOAT(6) ,
      PA ( 2 ) PTR,
        1 POINT_PA ( 2 ) BASED( PA ( 1 ) ) ,
          2 HANGER PTR,
          2 TYPE BIN FIXED(15),
          2 SYMBOL BIN FIXED(15),
          2 P_UP PTR,
          2 HEIGHT DEC FLOAT(6),
          2 X(3) DEC FLOAT(6) ,

/*   26*/    1 POINT_PB CONTROLLED ,
              2 PB PTR,
              2 TYPE BIN FIXED(15),
              2 SYMBOL BIN FIXED(15),
              2 P_UP PTR,
              2 HEIGHT DEC FLOAT(6),
              2 X(3) DEC FLOAT(6) INIT
              CALL GRCBINI(QQ,ADDR( POINT_PB ),   1 , 1 ,1) ,

/*   27*/    1 POINT_PC BASED ( P_PC ) ,
              2 PC PTR,
              2 TYPE BIN FIXED(15),
              2 SYMBOL BIN FIXED(15),
              2 P_UP PTR,
              2 HEIGHT DEC FLOAT(6),
              2 X(3) DEC FLOAT(6) INIT
              CALL GRCBINI(QQ,ADDR( POINT_PC ),   1 , 1 ,1) ,

/*   30*/CALL GRASIGN ( QQ , DREI ,
          QQ_GRPOLY_E(QQ_GRPOLY_P,QQ_GRCOLO6_E(QQ_GRCOLO6_P, 4,PS,
PA(1),PA(2),PS,NULL(),NULL())
          )
          ,'1'B);

/*   31*/P1 : PROCEDURE RETURNS ( POINTER ) ;
          RETURN (
          QQ_GRNPOI_E(QQ_GRNPOI_P,2,DREI
          )
          ) ;
          END P1 ;
```

Abb. 3.7: Auszug aus der zum Beispiel aus Abb. 3.6 generierten Liste

schen Bereich entspricht. Durch eine graphische Zuweisung werden die Attribute einer graphischen Variablen in das Zielobjekt übertragen. Da die graphischen Attribute entsprechend Abb.3.5 für die Objekte der Spracherweiterung nur durch Subroutinenaufruf übertragbar sind, muß eine Modifikation des Zuweisungsstatements erfolgen. Eine Anweisung

P1 = P2; ,

die die Gleichsetzung zweier Punktvariablen bewirken soll, könnte jedoch von Problemsprachensystem PLS nicht als Bestandteil der Spracherweiterung erkannt werden, da es sich um eine legale Zuweisung im PL/1-Sinne handelt und eine Auslese über die Namen und den graphischen Typ der Variablen nicht erfolgt (vgl.Abschnitt 3.3.1.2).

Zur Unterscheidung beginnt deshalb die graphische Zuweisung in GIPSY mit SET und lautet vollständig:

<graphical assignment>:: = SET <grevar> = <grex> ;

Die beiden variablen Sprachelemente ('nonterminals') in dieser Produktion stehen für:

<grevar> — graphical element variable:

Ein durch Namen (und Index) identifiziertes elementares graphisches Objekt.

<grex>::= <grop>|<grevar>

graphical element expression:

Ein graphischer Ausdruck bzw. eine Operation <grop>, die als Ergebnis ihrer Ausführung ein elementares graphisches Objekt liefert, oder eine 'graphical element variable' selbst.

Eine graphische Operation ist die zusammenfassende Bezeichnung für folgende Konstruktionen:

<grop>::= <grab>|<grat> |<graf-ref>|<grast-ref>

Die in dieser Definition verwendeten Abkürzungen bedeuten:

grab — graphical builtin function:

Eine zum Sprachumfang der GIPSY-Sprache gehörende Funktion, die als Ergebnis ihrer Ausführung in Abhängigkeit von geometrischen und/oder graphischen Argumenten ein neues elementares graphisches Objekt erzeugt.

grat — graphical transformation:

Eine zum Sprachumfang der GIPSY-Sprache gehörende Transformation, die als Ergebnis ihrer Ausführung ein in seinen geometrischen Attributen gegenüber dem Argument verändertes neues elementares graphisches Objekt erzeugt.

graf — graphical function:

Eine vom Anwender spezifizierbare Funktion, die als Ergebnis ihrer Ausführung ein elementares graphisches Objekt zurückliefert.

grast — graphical statement function:

Eine vom Anwender durch Anweisung DEFINE spezifizierbare Funktion, die als Ergebnis ihrer Ausführung ein elementares graphisches Objekt zurückliefert.

ref — Diese Angabe besagt, daß es sich um die Referenz auf eine der obengenannten Funktionen handelt.

Auf die Verwendung graphischer Operationen wird im nächsten Abschnitt näher eingegangen. Deshalb sollen hier zur Demonstration der graphischen Zuweisung nur Beispiele mit Elementvariablen erscheinen, wie

```
SET  PB = PA(2);
```

und

```
SET  PC = PS;
```

aus Statement 33 und 34 der Abb. 3.6, sowie

```
SET  OBJEKT.SUBOBJECT.A=B;
```

oder

```
SET  P → DREIECK(N).SEITE(K).PUNKT=ECKE(M);
```

Da die Deklaration von graphischen Objekten in PL/1-Strukturanweisungen umgesetzt werden (vgl. Abb. 3.7), ist es möglich, den verschiedenen Attributen eines Objektes explizit Werte zuzuweisen, wie dies in Statement 21 in Abb. 3.6 für die Koordinaten geschieht. Es muß bei dieser Zuweisung, die ausschließlich nach PL/1-Regeln erfolgt, darauf geachtet werden, daß die

Angaben zu geometrischen Abmessungen in der Längeneinheit Meter, in die alle Werte zur internen Behandlung umgerechnet werden, erfolgen.

Die Benutzer von Rechenprogrammen sind aus vielen Anwendungen mit einer Eingabetechnik vertraut, die die Angabe der Koordinatenwerte als Folge von Zahlentupeln (x,y(,z)) erwarten. Um diese Eingabe auch in GIPSY möglich zu machen, wurde das Statement

FILL <grevar> WITH <spec> [,<spec>]*;

geschaffen. Nach den Regeln

<spec>::= <elementexpr> [TO <elementexpr> [BY <elementexpr>] | BY <elementexpr> [TO <elementexpr>]]

lassen sich Listen verwenden, die

- Konstanten,
- Variablen,
- Ausdrücke und
- Schleifen

enthalten:

```
FILL  A.B(7) WITH  0.5,18.,-0.523,
                   X,SIN(X),Z**2,
                   5. TO 14. BY COS(X);
```

Diese Form der Eingabe kann für Polygonzüge (2D+3D) und Kurven in Problemkoordinaten herangezogen werden. Die vorgenannten graphischen Zuweisungen übernehmen die Übertragung von graphischer Information in existierende namentlich identifizierbare Objekte, die die Grundlage für die algorithmische Formulierung eines Problemes darstellen. Der Zugriff auf diese Objekte und die Erzeugung der Referenzen auf sie,wird vom Compiler der Basissprache übernommen.

Neben der Fähigkeit die graphische Information in dafür zugeschnittene Strukturen abzulegen, ist die Mächtigkeit einer Sprache von dem Umfang der zur Erzeugung und Manipulation der graphischen Daten bereitstehenden Operationen bestimmt.

3.3.3 Graphische Operationen

In den graphischen Operationen ist die semantische Erweiterung der Grundsprache realisiert. Sie ermöglichen den Umgang mit den graphischen Objekten in einer Weise, die sich an den geometrischen Vorstellungen und Wünschen orientiert. Für die Erzeugung graphischer Objekte aus

- arithmetischen und/oder
- anderen graphischen Objekten

werden etwa 50 Operationen angeboten. Einen Überblick zu den verfügbaren Operationen, die nach den Erfordernissen der Zeichenpraxis ausgwählt wurden, und den bei ihrer Anwendung erwarteten Argumenten gibt Abb. 3.8.

Nach der oben gegebenen Definition für die graphischen Builtin-Funktionen gilt danach:

Graphische Builtin-Funktionen

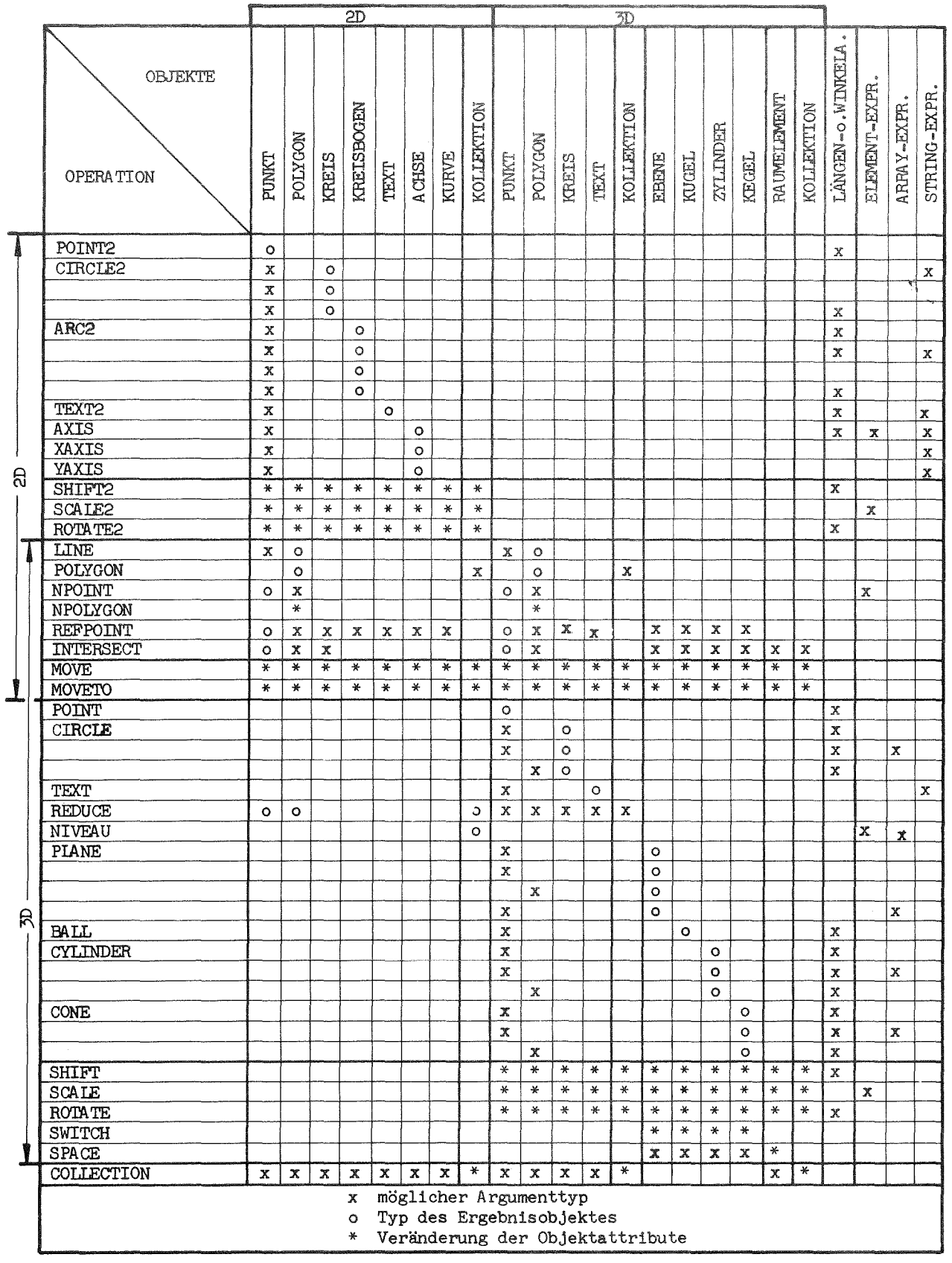
```
<grab>::= <point2>    |<circle2>  |<arc2>     |<text2>|
          <axis>      |<xaxis>    |<yaxis>    |
          <line>      |<polygon>  |<npoint>   |<npolygon>|
          <refpoint>  |<intersect>|
          <point>     |<circle>   |<text>     |
          <plane>     |<ball>     |<cylinder> |<cone>|
          <space>     |
          <collection>|<niveau>
```

Aus der Vielfalt der Funktionen sollen hier zur weiteren Erläuterung nur POINT, LINE,INTERSECT und COLLECTION herausgegriffen werden. Zur vollständigen Beschreibung der Syntax und Semantik wird auf Anhang A verwiesen.

Die Anwendung einer Builtin-Funktion zur Erzeugung einer Linie ist gegeben durch

```
<line>::=LINE(<grex>,<grex>)
```

Die syntaktischen Anforderungen an die Richtigkeit einer Anwendung sind erfüllt, wenn zur Funktion LINE zwei beliebige nach den Regeln für graphische Ausdrücke gebildete Argumente spezifiziert werden. Für eine sinnvolle Ausführung der Aufgabe ist jedoch gefordert, daß diese Argumente vom Typ POINT sind. Ein Statement zur Erzeugung einer Linie könnte demnach lauten:



	OBJEKTE / OPERATION	2D								3D													
		PUNKT	POLYGON	KREIS	KREISBOGEN	TEXT	ACHSE	KURVE	KOLLEKTION	PUNKT	POLYGON	KREIS	TEXT	KOLLEKTION	EBENE	KUGEL	ZYLINDER	KEGEL	RAUMELEMENT	KOLLEKTION	LÄNGEN-o.WINKELA.	ELEMENT-EXPR.	ARRAY-EXPR.	STRING-EXPR.
2D	POINT2	o																			x			
	CIRCLE2	x		o																				x
		x		o																				
		x		o																	x			
	ARC2	x			o																x			
		x			o																x			x
		x			o																			
		x			o																x			
	TEXT2	x				o															x			x
	AXIS	x					o														x	x		x
	XAXIS	x					o																	x
	YAXIS	x					o																	x
	SHIFT2	*	*	*	*	*	*	*	*												x			
	SCALE2	*	*	*	*	*	*	*	*													x		
	ROTATE2	*	*	*	*	*	*	*	*												x			
	LINE	x	o							x	o													
	POLYGON		o						x		o			x										
	NPOINT	o	x							o	x											x		
	NPOLYGON		*								*													
	REFPOINT	o	x	x	x	x	x	x		o	x	x	x		x	x	x	x						
	INTERSECT	o	x	x						o	x				x	x	x	x	x	x				
	MOVE	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*				
	MOVETO	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*				
3D	POINT									o											x			
	CIRCLE									x		o									x			
										x		o									x		x	
											x	o									x			
	TEXT									x			o											x
	REDUCE	o	o						o	x	x	x	x	x										
	NIVEAU								o													x	x	
	PLANE									x					o									
										x					o									
											x				o									
										x					o								x	
	BALL									x						o					x			
	CYLINDER									x							o				x			
										x							o				x		x	
											x						o				x			
	CONE									x								o			x			
										x								o			x		x	
											x							o			x			
	SHIFT									*	*	*	*	*	*	*	*	*	*	*	x			
	SCALE									*	*	*	*	*	*	*	*	*	*	*		x		
	ROTATE									*	*	*	*	*	*	*	*	*	*	*	x			
	SWITCH														*	*	*	*						
	SPACE														x	x	x	x	*					
	COLLECTION	x	x	x	x	x	x	x	*	x	x	x	x	*					x	*				

x möglicher Argumenttyp
o Typ des Ergebnisobjektes
* Veränderung der Objektattribute

Abb. 3.8: Tabelle der graphischen Operationen

```
SET  L(N)=LINE(P1,POINT(X,Y,Z**2  CM));
```

In diesem Beispiel ist das als Argument erwartete Punktobjekt, zum einen als graphische Elementvariable, zum anderen als graphische Funktion angegeben. Wie diese Anweisung außerdem zeigt, können Angaben zu Längen (und Winkeln) durch mit Einheiten behaftete Ausdrücke erfolgen.

Die syntaktische Verarbeitung der Anweisungen mit graphischen Ausdrücken erfolgt nach der Methode des rekursiven Abstieges, um eine hohe Flexibilität in dem Formulierungsmöglichkeiten zu gewährleisten. Die daraus entstehenden Fähigkeiten sollen am folgenden Beispiel deutlich gemacht werden, in dem die Funktion INTERSECT zur Schnittpunktbestimmung zwischen zwei Linien verwendet wird:

```
SET  L=LINE(P(N),INTERSECT(LI,LINE(P,POINT(X,Y,Z))));
```

Die Unterstreichungen in diesem Statement machen die Levels der Rekursivität deutlich, indem jeder Strich einen gültigen graphischen Ausdruck hervorhebt.

Die Anweisungen 8,19,20,30,31 und 42 der in Abb. 3.6 gezeigten Liste enthalten graphische Builtin-Funktionen im Kontext eines Programmes, während in Abb. 3.9 eine Gegenüberstellung von spezifizierender Eingabe und resultierender graphischer Darstellung der Strichobjekte vorgenommen wird.

Die Flexibilität der graphischen Zuweisungen beruht vor allem auf:

- der Beibehaltung von PL/1-Eigenschaften, wie Datenfelder, arithmetische Ausdrücke, usw., und
- der Fähigkeit zur Erzeugung von graphischen Objekten durch verschiedenartige Argumentkombinationen.

Beispielhaft für alle anderen Objekte sei die Vielfalt der Angaben zur Erzeugung eines zweidimensionalen Kreisbogens genannt, der durch

- einen Punkt, zwei Winkel und Radius,
- einen Punkt und Länge des Bogens,
- zwei Punkte, Radius und die Wahl der großen oder kleinen Verbindung und
- drei Punkte

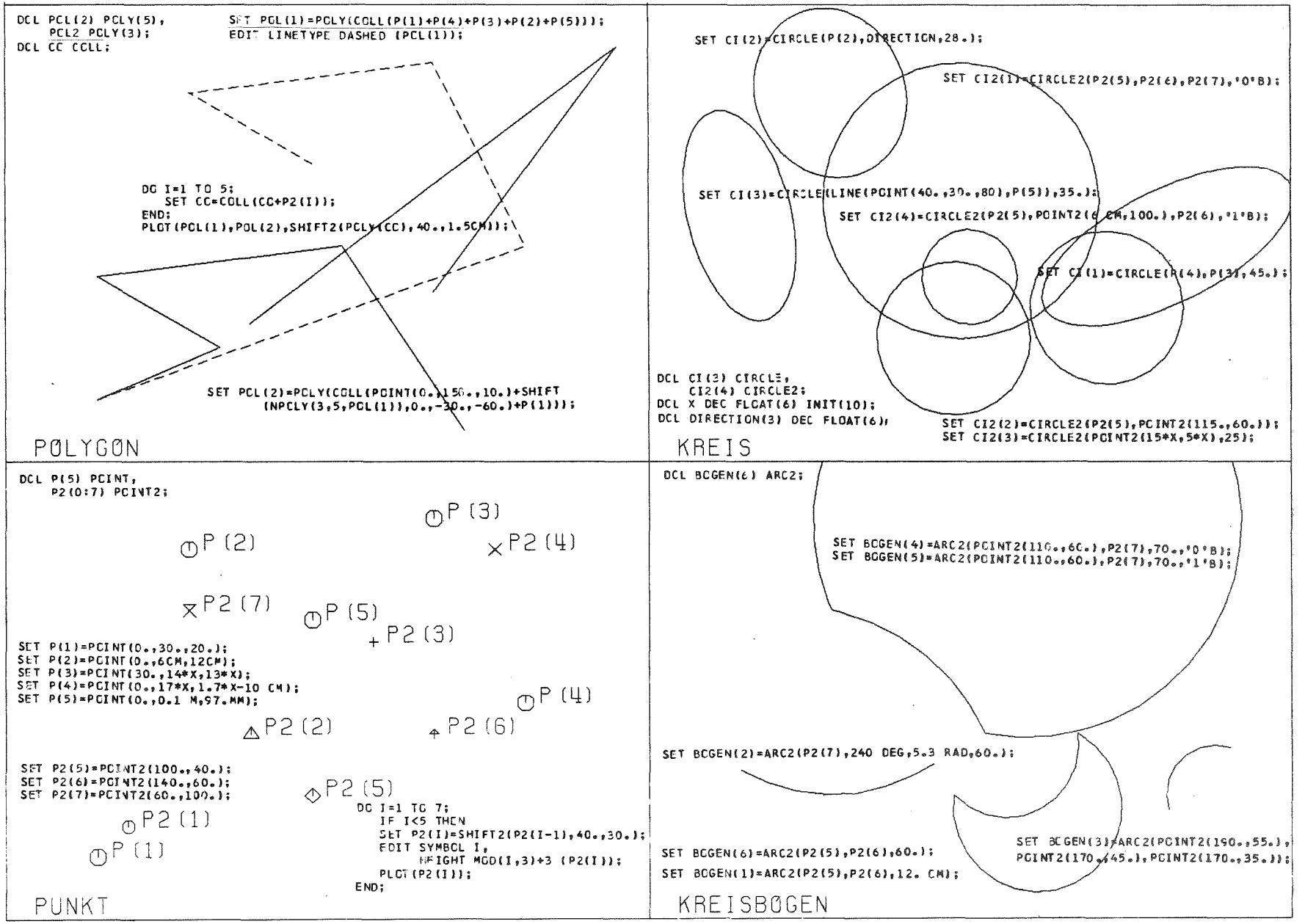


Abb. 3.9: Demonstration der Möglichkeiten einiger graphischer Operationen

spezifiziert werden kann (vgl. Abb. 3.8 und 3.9).

Bei der Ausführung derartiger Operationen ist die Erzeugung der gewünschten graphischen Objekte aufgrund fehlerhafter Argumentübergabe in manchen Fällen nicht möglich. Um einen Abbruch der Rechnung zu verhindern, wird dann ein undefiniertes Objekt zurückgeliefert, das alle graphischen Routinen akzeptieren. Die übrige Rechnung kann dann unter Vernachlässigung der fehlerhaften Objektspezifikationen ausgeführt werden, wodurch in der Regel eine Verkürzung der Gesamttestzeit für ein Programm erzielt wird.

Bei graphischen Aufgaben stellt sich bei vielen Anwendungen das Problem, die in einem Bild enthaltenen Beziehungen auch in der rechnerinternen Datenstruktur zu repräsentieren. Durch die Verwendung von Namen zur Identifikation von graphischen Objekten wird die Realisierung des Zugriffes vom Compiler übernommen und so eine aufwendige Zeigerverwaltung zur Ausführungszeit vermieden. Die Zusammenfassung von graphischen Objekten zu größeren Einheiten, die eine effektivere Beschreibung der graphischen Aufgabe erlauben, ist eine allgemein gestellte Forderung, die in GIPSY auf drei Wegen unterstützt wird:

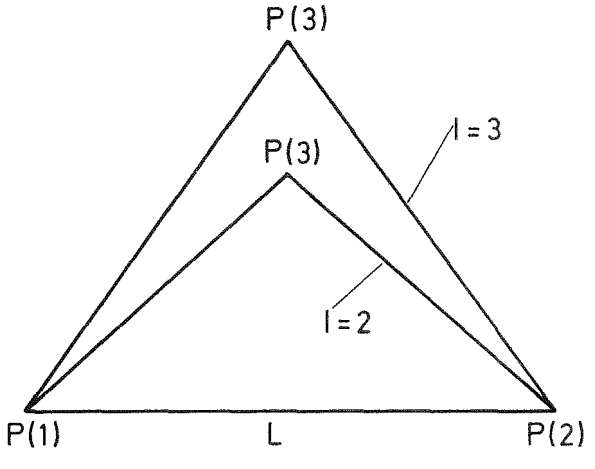
- Durch Feldvereinbarungen und Schleifenprogrammierung ist es möglich, viele Probleme algorithmisch zu schreiben, wobei allerdings die Objekte des Feldes auf einen Typ beschränkt sind.
- Die Definition und Anwendung von Prozeduren zur Beschreibung gleichartig strukturierter Objekte bzw. deren Abbildungen in Abhängigkeit von graphischen und/oder arithmetischen Argumenten oder globalen Variablen. Auf dieser Ebene bietet GIPSY das adaequate Werkzeug zur Vereinbarung von Bauteilbibliotheken an, da für diese Tätigkeit alle algorithmischen Fähigkeiten von PL/1 mit einem um graphische Datentypen erweiterten Sprachumfang herangezogen werden können. Die formale Vereinbarung und der Aufruf dertiger Routinen ('graphical functions') wird anhand der Statements 23 und 22 bzw. 37 und 35 des Beispiels in Abb. 3.6 ersichtlich. Auf die Möglichkeit der Verwendung externer und bereits übersetzter externer Routinen wird später noch eingegangen werden.

- Die Zusammenfassung von graphischen Objekten unterschiedlichen Typs auf dem Statementlevel wird durch Kollektionen erreicht. Eine Kollektion wird erzeugt durch die Angabe der Liste derjenigen Objekte, die in ihr aufgesammelt werden sollen, um dadurch über einen Namen ansprechbar zu sein. Es wird dabei jedoch nicht das Objekt selbst, sondern eine temporäre Kopie mit gleichen graphischen Attributen eingebaut.

Die Fähigkeiten einer Kollektionsbildung sollen an einem einfachen Beispiel demonstriert werden:

```
DCL   P(3)     POINT2,
      L        POLY2(2),
      DREIECK COLLECTION;
SET   P(1) = POINT2(-1.,0.);
SET   P(2) = POINT2(1.,0.);
SET   L = LINE(P(1),P(2));
DO I  = 3,2;
      SET P(3)=POINT2(0.,I);
      SET DREIECK = COLLECTION
          (L+LINE(P(1),P(3))+LINE(P(2),P(3)));
      PLOT(DREIECK);
END;
```

Diese Statementfolge bewirkt die Ausgabe:



Eine einfache aber elegante Möglichkeit zum Auffädeln von Objekten in einer Kollektion ergibt sich aus der Heranziehung einer Schleifenvereinbarung:

```
DCL     ALLES COLLECTION;
DO  I = 1  TO 10;
    SET    ALLES=COLL(ALLES+OBJECT(I));
END;
```

Graphische Transformationen

Die graphischen Transformationen beeinflussen gemäß Definition die geometrischen Attribute eines Objektes. Von den in Abb. 3.8 aufgelisteten Operationen fallen unter diese Kategorie:

```
<grat>::= <shift2>|<scale2>|<rotate2>|
          <move>  |<moveto>|
          <reduce>|
          <shift> |<scale> |<rotate>|
          <switch>|
```

Mittels dieser Funktionen stehen einem Anwender alle Möglichkeiten bereit, die eine lineare Transformation in zwei und drei Dimensionen bietet, so daß die beim Positionieren oder Anpassen von Teilabbildungen auftretenden Probleme leicht programmierbar sind.

Im Gegensatz zu vielen anderen Systemen sind diese Transformationen auf einzelne Objekte unabhängig anwendbar und die Ergebnisse ihrer Ausführung in den Objekten abspeicherbar, d.h. die Transformationen kommen nicht erst bei der Plotausgabe zur Anwendung, sondern unterstützen bei der algorithmischen Formulierung der Aufgabe. Graphische Transformationen können wie Builtin-Funktionen geschachtelt aufgerufen werden.

Für die gemischte Anwendung von zwei- und dreidimensionaler Strichgraphik stellt die Operation REDUCE eine besondere Fähigkeit bereit. Sie ermöglicht die Zuweisung von reduzierten - d.h. unter Anwendung der aktuellen Projektionsmatrix abgebildeten - 3D-Objekten in ihre 2D-Repräsentanten. Durch diese Hilfe ist ein'Anbinden' von zweidimensionaler graphischer Information an die Bildkoordinaten dreidimensionaler Objekte möglich.

Graphische Statement-Funktion

Durch eine graphische Statement-Funktion wird der funktionale Zusammenhang zwischen einem Namen und einem graphischen Ausdruck hergestellt. Zum Zeitpunkt des Referierens wird mit den dann jeweils gültigen Argumenten die Funktion zur Ausführung gebracht. Die Statements 31 und 36 in Abb. 3.6 zeigen ein Beispiel für die Anwendung dieser Fähigkeit, die sich vom graphischen Assignment durch das Schlüsselwort DEFINE unterscheidet.

In diesem Teil der Beschreibung der Fähigkeiten wird nicht auf die Operationen zur Erzeugung der räumlichen Objekte Flächen, Raumelemente und Körper eingegangen, da vor deren Anwendung im nächsten Abschnitt eine Erläuterung der systemeigenen Behandlung räumlicher Information gegeben werden soll.

Im vorliegenden Kapitel wurde auch darauf verzichtet, größere Anwendungsbeispiele für die bereitgestellten graphischen Operationen zu bringen.

Die Vielfalt der graphischen Objekte und Operationen (vgl. Abb. 3.8) ist auf die Bedürfnisse der zeichnerischen Aufgaben eines Konstrukteures zugeschnitten. Sie stellen neben den noch zu behandelnden Fähigkeiten zur Körperbehandlung ein universell einsetzbares Werkzeug bei der Formulierung graphischer Aufgaben dar.

3.3.4 Graphische Ausgabe und deren Kontrolle

Das System GIPSY unterstützt in der vorliegenden Version nur die Ausgabe von graphischer Information. Die Eingabe der graphischen Information erfolgt durch das Problemprogramm oder durch die Fähigkeiten der Basissprache PL/1 zum Einlesen von modellbeeinflussenden Parametern. Die Beschreibung der Ausgabestatements erfolgt entsprechend der in Abb. 3.3 vorgenommenen Einteilung.

3.3.4.1 Druckausgabe des System- und Objektzustandes

Zur Kontrolle oder Protokollierung der graphischen Aufgabe sind die Statements STATUS und PRINT geeignet:

STATUS...; listet den aktuellen Zustand der Systemstruktur zu
- den Standardlängen- und -winkelangaben,
- der Koordinatenbasis,
- den Abbildungsparametern,
- den Inkrementen bei der Kurvenberechnung,
- den Linientypen und/oder
- dem ausgenutzten Papierbereich.

Eine beispielhafte Ausgabe, die den Anfangszustand einer Programmausführung charakterisiert, zeigt Abb. 3.10.

Die Attributwerte graphischer Daten sind in einer Datenstruktur gespeichert. Ein Anwender des Systems könnte direkt die Fähigkeiten von PL/1 nutzen, um die ihn interessierenden Werte auf dem Drucker darzustellen. Das System erleichtert diese Ausgabe der graphischen Objekte durch den PRINT-Befehl:

PRINT (<grex>[,<grex>]*);
bewirkt die Ausgabe aller in der Liste aufgeführter graphischer Elemente unter Angabe
- des Namens,
- des spezifizierten Index und
- aller graphischen Attribute des Objektes.

In Abb. 3.11 wird das Ergebnis der Ausführung des Beispiels aus Abb. 3.6 gezeigt. Die Ausgabe erscheint in einer leicht lesbaren, strukturierten Form als Liste auf dem Schnelldrucker und kann zu Testzwecken an die Stelle der noch zu erläuternden graphischen Ausgabe treten oder diese auch nur ergänzen.

```
+-------------------------------------------------------------------+
|                                                                   |
| GRAPHIC STATE INFORMATION: ALL                                    |
|                                                                   |
+-------------------------------------------------------------------+
| LENGTH UNIT   |   MM                                              |
| ANGLE UNIT    |   DEGREES                                         |
+-------------------------------------------------------------------+
| COORDINATE VALUES ARE ASSUMED TO BE ABSOLUTE                      |
+-------------------------------------------------------------------+
| PARALLEL PROJECTION WITH DIRECTION:                               |
|  1.000000E+00   0.000000E+00   0.000000E+00                       |
+-------------------------------------------------------------------+
| PICTURE PLANE:                                                    |
+-------------------------------------------------------------------+
| PI_N          |   1.000000E+00   0.000000E+00   0.000000E+00      |
| PI_P          |   0.000000E+00   0.000000E+00   0.000000E+00      |
| D             |   0.000000E+00                                    |
+-------------------------------------------------------------------+
| PROSPECTIVE MATRIX:                                               |
+-------------------------------------------------------------------+
|  1.000000E+00   0.000000E+00   0.000000E+00   0.000000E+00        |
|  0.000000E+00   0.000000E+00   1.000000E+00   0.000000E+00        |
|  0.000000E+00   0.000000E+00   0.000000E+00   1.000000E+00        |
+-------------------------------------------------------------------+
| INCREMENT FOR COMPUTATION OF INTERSECTIONS:                       |
+-------------------------------------------------------------------+
| STRAIGHT(M)   |   3.999997E-03                                    |
| CURVED(DEC)   |   5.000000E+00                                    |
+-------------------------------------------------------------------+
| ORIGIN(M)     |   0.000000E+00   0.000000E+00                     |
| SCALE         |   1.000000E+00   1.000000E+00                     |
| ROTATION(DEG)|   0.000000E+00                                    |
+-------------------------------------------------------------------+
|   VISIBLE LINE  ----------                                        |
| INVISIBLE LINE  - - - - -                                         |
|    CENTER LINE  -.-.-.-.-.                                        |
+-------------------------------------------------------------------+
| DOWNLEFT(M)   |   1.000000E+50   1.000000E+50                     |
| UPRIGHT(M)    |  -1.000000E+50  -1.000000E+50                     |
+-------------------------------------------------------------------+
| PAPER(M)      |   5.000000E+00   5.300000E-01                     |
| FIGURE(M)     |   2.100000E-01   2.970000E-01                     |
+-------------------------------------------------------------------+
| SORT OF PAPER|   1                                                |
| SORT OF PEN   |   1                                               |
+-------------------------------------------------------------------+
```

Abb. 3.10: Ausgabe der Systeminformation

```
+-----------------------------------------------------------------+
| POLYGON:SHIFT                                                   |
+-----------------------------------------------------------------+
| LINETYPE:OMITTED          | EVERY   1.POINT OPEN                |
| NO. OF POINTS:    3       | LENGTH: 4.999999E-03                |
+-----------------------------------------------------------------+
| SYMBOL:    1              | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| NO.    |       X(1)              X(2)              X(3)          |
+-----------------------------------------------------------------+
|      1 |   9.999990E-02      2.000000E-03      2.999999E-03      |
|      2 |   9.999990E-02      2.000000E-03      2.999999E-03      |
|      3 |   9.999990E-02      2.000000E-03      2.999999E-03      |
+-----------------------------------------------------------------+
| THIS WAS A TEMPORARY OBJECT                                     |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| POINT:PE                                                        |
+-----------------------------------------------------------------+
| SYMBOL:    1              | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| X(*)   | 1.999998E-01      4.999999E-02      5.761999E-02          |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| TEXT:STR                                                        |
+-----------------------------------------------------------------+
| NO. OF CHAR.:   10        | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| X1(*)  | 9.999998E-03      9.999996E-02      1.999999E-01          |
| X2(*)  | 1.999998E-01      4.999999E-02      5.761999E-02          |
+-----------------------------------------------------------------+
| STRING:ZURUECK AU                                               |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| POINT:PF(1)                                                     |
+-----------------------------------------------------------------+
| SYMBOL:    1              | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| X(*)   | 9.999998E-03      9.999996E-02      1.999999E-01          |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| POINT:PF(2)                                                     |
+-----------------------------------------------------------------+
| SYMBOL:    1              | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| X(*)   |-7.000002E-04      7.141421E-04      0.000000E+00          |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| POLYGON:DREI                                                    |
+-----------------------------------------------------------------+
| LINETYPE:OMITTED          | EVERY   1.POINT OPEN                |
| NO. OF POINTS:    4       | LENGTH: 4.999999E-03                |
+-----------------------------------------------------------------+
| SYMBOL:    1              | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| NO.    |       X(1)              X(2)              X(3)          |
+-----------------------------------------------------------------+
|      1 |   4.999999E-02      4.999999E-02      4.999999E-02      |
|      2 |   9.999998E-03      9.999996E-02      1.999999E-01      |
|      3 |  -7.000002E-04      7.141421E-04      0.000000E+00      |
|      4 |   4.999999E-02      4.999999E-02      4.999999E-02      |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| POINT:P1                                                        |
+-----------------------------------------------------------------+
| SYMBOL:    1              | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| X(*)   | 9.999998E-03      9.999996E-02      1.999999E-01          |
+-----------------------------------------------------------------+
| THIS WAS A TEMPORARY OBJECT                                     |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| CIRCLE:OBJECT.SUBOBJECT.A(2)                                    |
+-----------------------------------------------------------------+
| CENTER |-7.000002E-04      7.141421E-04      0.000000E+00        |
| AXIS   | 7.000007E-01     -7.141426E-01      0.000000E+00          |
+-----------------------------------------------------------------+
| RADIUS: 9.999997E-06                                            |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| TEXT:T3                                                         |
+-----------------------------------------------------------------+
| NO. OF CHAR.:   15        | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| X1(*)  | 9.999998E-03      9.999996E-02      1.999999E-01          |
| X2(*)  | 1.999998E-01      4.999999E-02      5.761999E-02          |
+-----------------------------------------------------------------+
| STRING:ZURUECK AU                                               |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| POINT:P5                                                        |
+-----------------------------------------------------------------+
| SYMBOL:    1              | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| X(*)   | 4.999999E-02      4.999999E-02      4.999999E-02          |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| POINT:PB                                                        |
+-----------------------------------------------------------------+
| SYMBOL:    1              | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| X(*)   |-7.000002E-04      7.141421E-04      0.000000E+00          |
+-----------------------------------------------------------------+

+-----------------------------------------------------------------+
| POINT:PC                                                        |
+-----------------------------------------------------------------+
| SYMBOL:    1              | HEIGHT: 4.999999E-03                |
+-----------------------------------------------------------------+
| X(*)   | 1.999998E-01      4.999999E-02      5.761999E-02          |
+-----------------------------------------------------------------+
```

Abb. 3.11: Druckausgabe graphischer Objekte (nach Abb. 3.6)

3.3.4.2 Kontrolle des System- und Objektzustandes

Zur Erzielung einer hohen Flexibilität sind alle System- und Objektzustände durch Mittel der GIPSY-Sprache, die hierzu die Befehle CHANGE und EDIT kennt, veränderbar.

CHANGE...; bestimmt neue Werte für die Systemzustände zu
- den Standardeinheiten (CM,RAD,..),
- der Koordinatenbasis (relativ|absolut),
- der Projektionsart (parallel|zentral,..),
- der Linienart (z.B. für unsichtbare Kanten (——,---,), usw.

EDIT...,...(<grex>[,<grex>]*);
verändert die objektspezifischen Ausgabezustände der graphischen Elemente, wie
- den Symboltyp von Punkten,
- die Schrifthöhe von Texten und Punktsymbolen,
- die Linienart von Polygonzügen und deren Kennzeichnungen und
- die Charakteristik von Koordinatenachsen.

Eine Anweisung, die den Systemzustand beeinflußt, ist als gesondertes Statement verfügbar, da sie eine ausgezeichnete Wirkung auf Ausgabe der graphischen Information besitzt.

OPEN PLOT [DIN A <elementexpr> [BROAD]] ;
[<lvalue> x <lvalue>]

Durch diese Angabe wird die für die folgenden Ausgabeoperationen verfügbare Zeichenfläche festgelegt und eröffnet, wobei auf eine optimale Anordnung dieser Zeichenrahmen hinsichtlich Papierverbrauch geachtet wird.

3.3.4.3 Graphische Ausgabe

Alle Aktivitäten zur graphischen Ausgabe graphischer Information gehen vom PLOT-Statement aus, das vier Varianten besitzt:

PLOT (< grex> [,< grex>]*);

In dieser Form bewirkt PLOT die zu zeichnende Darstellung der graphischen Objekte in der Liste unter Beachtung der Angaben zum System- und Objektzustand. Bei Kollektionen wirkt diese Anweisung auf alle Subobjekte, 3D-Objekte werden entsprechend der gültigen Abbildungsvorschrift projiziert und Raumelemente bzw. Körper werden in der noch zu beschreibenden Weise analysiert und dann deren Kanten gezeichnet.

Bei der Ausgabe werden alle Werte auf das Verlassen der Zeichenfläche überprüft ('scissoring').

PLOT DIAGRAMME WITH [OUT] [AXIS] < grevar> [AND] < grevar>
OF (< grevar> [,< grevar>]*);

Diese Variante unterstützt die Ausgabe von Kurven in Problemkoordinaten, wie sie die CURVE-Objekte enthalten. Sie deckt denjenigen Bereich, den die Darstellung von Wertefeldern aus Experimenten und Rechnungen in Diagrammen benötigt, ab und macht ihn einer einfachen Anwendung zugänglich.

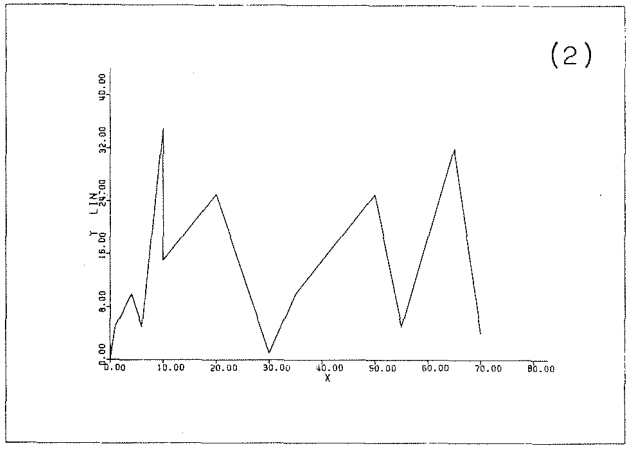
Die Kurven in Problemwerten werden entsprechend der angegebenen Koordinatenachsenpaarung umgerechnet. Für die Erzeugung der Achsen stehen besondere Funktionen zur Verfügung, die eine automatisierte Bestimmung passender Größen in Abhängigkeit von verfügbarem Bildfenster und Problemwertebereich erlauben. In Abb. 3.12 ist das Programm einiger Diagrammbeispiele seiner Ausgabe gegenübergestellt.

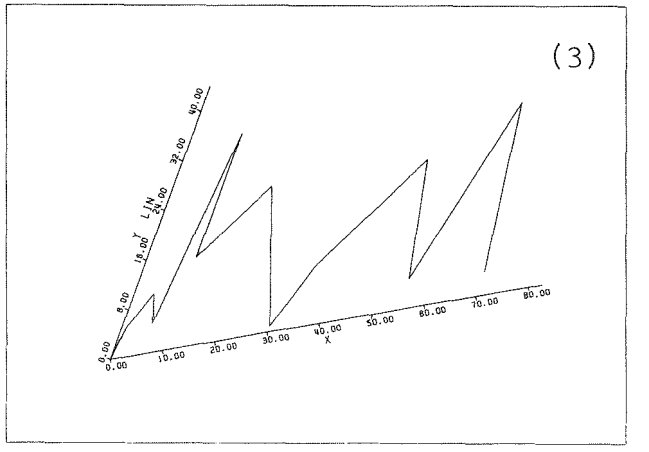
```
ACHSEN: PROC OPTIONS(MAIN) REGENT(NODA,POOL=100000,PLOT);
ENTER GIPSY;
/*****************************************************************/
        DCL DEGREE DEC FLOAT(6);
        DCL COSIN(2) CURVE(19),
            (XA,YA) AXIS(6);
          OPEN PLOT DIN A4 BROAD;
          DO I=1 TO 19;
            DEGREE=20.*(I-1);
            CURVE_COSIN(*).X(I,1)=DEGREE;
            CURVE_COSIN(1).X(I,2)=COSD(DEGREE);
            CURVE_COSIN(2).X(I,2)=SIND(DEGREE);
          END;
          SET XA=AXIS(POI2(30 MM,106.5),0.,360.,200.,0.,'X','LIN');
          SET YA=YAXIS(COSIN(1),'F(X)','LIN');
          EDIT LINETYPE DASHED (COSIN(1));
          EDIT LINETYPE MARKED,SYMBOL 12,EVERY 4,HEIGHT 3MM (COSIN(2));
        || PLOT DIAGRAMME WITH AXIS XA AND YA OF (COSIN(1),COSIN(2));
/*****************************************************************/
        DCL ZAEHNUNG CURVE(13),
            LIN(2) AXIS(6);
        DCL (XL INIT(30.),
             WINKEL) DEC FLOAT(6);
          FILL ZAEHNUNG WITH 0.,0.,1.,5.,4.,10.,6.,5.,10.,35.,
          10. TO 26. BY 5.,XL,1.,35.,10.,50.,25.,55.,5.,65.,32.,70.,4.;
          DO WINKEL=0.,10.;
            OPEN PLOT DIN A4 BROAD;
            SET LIN(1)=AXIS(POI2(50.,40.),0.,70.,21 CM,WINKEL,'X','');
            SET LIN(2)=AXIS(POINT2(50.,40.),0.,40.,14 CM,
                            (90.-2*WINKEL),'Y  LIN','LIN');
            EDIT LEFT (LIN(2));
        ||  PLOT DIAGRAMME WITH LIN(1) AND LIN(2) OF (ZAEHNUNG);
          END;
/*****************************************************************/
        DCL X_AX AXIS(1),
            Y_AX AXIS(19),
            EFUNC(2) CURVE(13);
          OPEN PLOT DIN A4 BROAD;
          DO J=1,2;
            DO I=-6 TO 6;
              CURVE_EFUNC(J).X(I+7,1)=(I/2.)+(J-1);
              CURVE_EFUNC(J).X(I+7,2)=1.E4*J*EXP(-(I/2.+(J-1))**2);
            END;
            EDIT SYMBOL J*2,EVERY J,LINE MARKED,HEIGHT 2 (EFUNC(J));
          END;
          SET X_AX=XAXIS(COLL(EFUNC(1)+EFUNC(2)),'X','LIN');
          SET Y_AX=AXIS(POINT2(131.,30.),1.E-3,1.E5,15 CM,90 DEG,
                        '1.E4*J*EXP(-(X)**2','LOG');
          EDIT LEFT (Y_AX);
        || PLOT DIAGRAMME WITH X_AX AND Y_AX OF (EFUNC(1),EFUNC(2));
          PLOT(TEXT2('VERTEILUNG', POINT2(30.,170.),0.));
/*****************************************************************/
END GIPSY;
FINISH;
END ACHSEN;
```

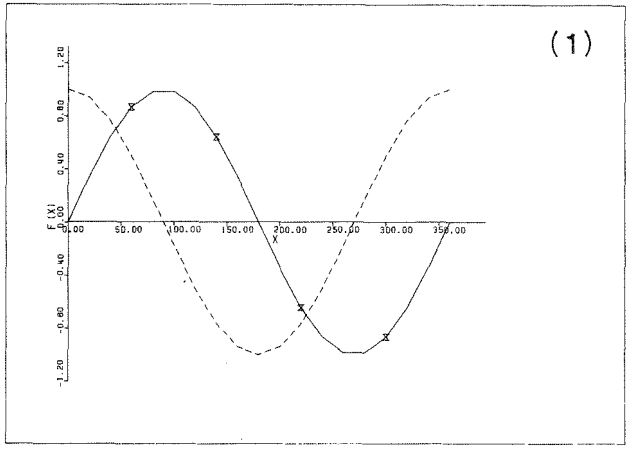
(1)

(2,3)

(4)







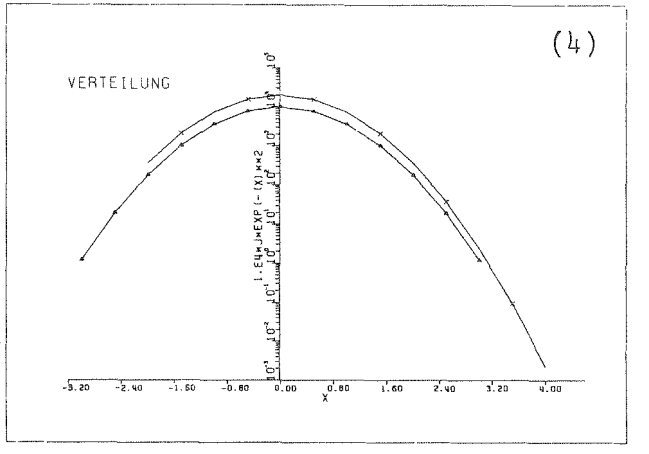


Abb. 3.12: Diagrammerstellung mit den Fähigkeiten zur Behandlung von Kurven in Problemkoordinaten

PLOT RELIEF [OF] <ident>[(<index>[,<index>]*)[.]]*...;

Im Gegensatz zum vorigen Statement, das die Ausgabe von Problemwertepaaren erlaubte, ist mit der Option RELIEF die perspektivische Darstellung von Problemwertetripeln unter Anwendung eines einfachen Sichtbarkeitskriteriums möglich. Über die Verdeckung wird ausschließlich in der Bildebene entschieden, da es sich um ein Verfahren handelt, das nur die Grenzen eines Sichtbarkeitsfeldes verwaltet und jede neu zu zeichnende Kurve gegen diese prüft und dann die Berandung aktualisiert. Das Verfahren basiert auf einer Arbeit von S.L.Watkins /170/, wurde jedoch hinsichtlich seiner Anwendung bei beliebiger Projektion, variabler Feldgrenzen und Verbesserung der 3D-Wirkung bei der Darstellung durch Hinzufügung von Randlinien verbessert.

Die Ausgabe erfolgt durch den gestaffelten Aufruf von Querschnitten durch das darzustellende Feld.

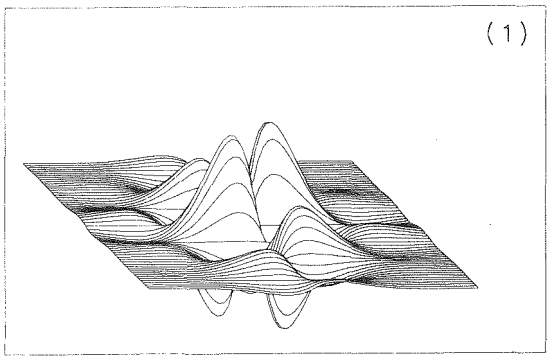
In der obigen Definition gilt:

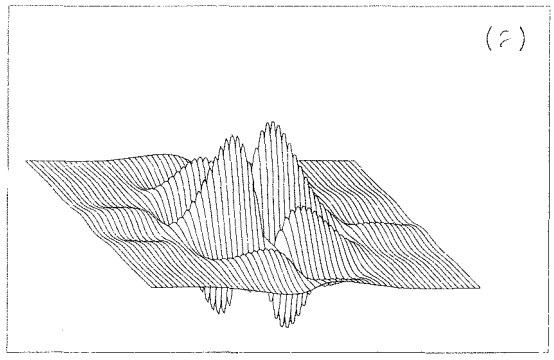
<index>:: = * | ** | <elementexpr >.

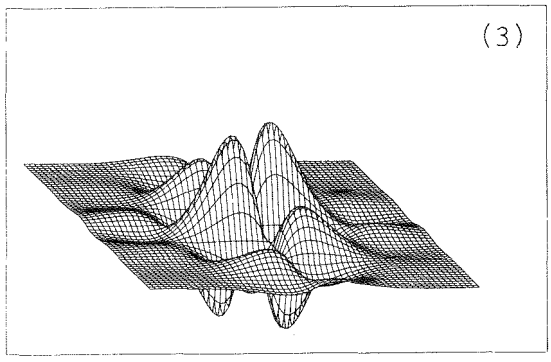
Mit der Vereinbarung des Doppelsternes (**) an einer Indexposition wird angedeutet, welcher Feld-Index automatisch von der unteren zur oberen Grenze variiert werden soll, um so die Eingabe komfortabler zu gestalten. Die Abb. 3.13 zeigt vier Anwendungen von PLOT RELIEF, wobei auch die Darstellung einer Fläche durch ein Netz von Rasterlinien angewendet wird.

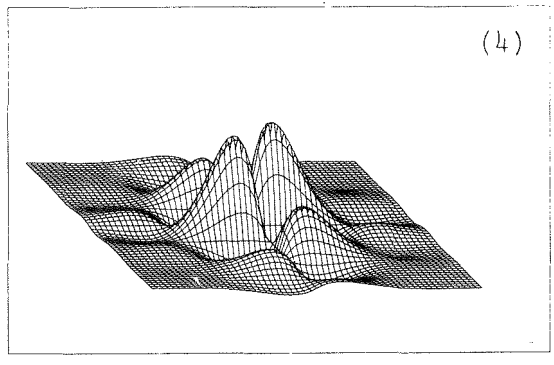
PLOT SHADE....;

Mit Hilfe dieser Anweisung können Schnitte durch räumliche Objekte spezifiziert und deren Darstellung mit Schraffur veranlaßt werden. Eine genaue Erläuterung dieser Fähigkeit wird unter Abschnitt 4.7 und im Anhang A vorgenommen.










```
 ALGO: PROC OPTIONS(MAIN) REGENT(NODA,PLOT);
ENTER GIPSY;
      DCL DAEMPFUNG BIT(1) INIT('1'B);
      DCL FELD(61) POLY(61);
 CHANGE PROJ ORIGIN 200.,90.,
        PROJ PARA 1.,-1.,1.;
          DO NP1=1 TO 61;
            BEAMV=SINDEXP(15.,NP1,2,30);
            POLYGON_FELD(NP1).X(*,1)=0. - 0.003 * (NP1-1);
            DO NP2=1 TO 61;
              POLYGON_FELD(NP1).X(NP2,2)=0.00254*3*(NP2-1);
              POLYGON_FELD(NP1).X(NP2,3)=0.5*BEAMV*
                                SINDEXP(7.5,NP2,2,20);
            END;
          END;
 OPEN PLOT 750. X 480.;
        PLOT REL X(**,*,*);          ------------------>  (1)
 OPEN PLOT 750. X 480.;
        PLOT REL X(*,**,*);          ------------------>  (2)
 OPEN PLOT 750. X 480.;
        PLOT REL X(**,**,*);         ------------------>  (3)
 OPEN PLOT 750. X 480.;
        PLOT REL X(**,**,*) LINES 'ABOVE';  ---------->  (4)
/*****************************************************************/
SINDEXP:PROC(A,N,M,L) RETURNS(DEC FLOAT(6));
      DCL (A,B) DEC FLOAT(6);
      DCL (K,L,M,N) BIN FIXED(15);
      K=3*N-93;
      B=SIN(A*SIND(K/M));
      IF DAEMPFUNG THEN
         B=B*EXP(-ABS(K/L));
         RETURN(B);
END SINDEXP;
/*****************************************************************/
  END GIPSY;
  FINISH;
 END ALGO;
```

Abb. 3.13: Perspektivische Darstellung von Wertefeldern

3.4 Verfügbarkeit der graphischen Fähigkeiten im REGENT-System

3.4.1 Aktivierung der Subsystemsprache

Durch die Angabe der Anweisung

ENTER GIPSY;

wird in einem REGENT-Programm die Aktivierung der Subsystemsprache erreicht und das für die Durchführung der graphischen Aufgaben erforderliche Environment etabliert. Die Fähigkeiten der problemorientierten Sprache bleiben bis zum Antreffen des Abschlußstatements

END GIPSY;

verfügbar.

Innerhalb dieses Bereiches, der einem Block in PL/1 entspricht, werden die subsystemspezifischen Anweisungen entsprechend den bei der Subsystemerstellung festgelegten Regeln in gültige PL/1-Anweisungen übersetzt.

3.4.2 Datenaustausch zwischen Subsystemen innerhalb eines Hauptprogrammes

In einem Programmlauf können Problemstatements mehrerer Subsysteme Verwendung finden. Diese Fachsprachen für verschiedene Aufgabengebiete sind jeweils durch entsprechende Anweisungen:

ENTER subsystem; und END subsystem;

eingeschlossen.

Die Subsystemaktivierung kann dabei geschachtelt oder hintereinander erfolgen. In beiden Fällen ist eine Datenübergabe mittels globaler Variabler möglich:

```
DCL  V1;
berechne: V1
ENTER  A;
  DCL  V2;
  berechne: V2
  ENTER  B;
    verwende: V1,V2
  END  B;
END  A;
```

```
DCL  V;
ENTER  A;
berechne: V
END  A;
ENTER  B;
verwende: V
END  B;
```

Neben dieser Kommunikation zwischen Subsystemen über globale Variable, wird eine Weitergabe von Werten z.B. auch über Files, Argumente von internen Prozeduren oder Datenbanken unterstützt.

3.4.3 Datenaustausch bei Anwendung externer Routinen

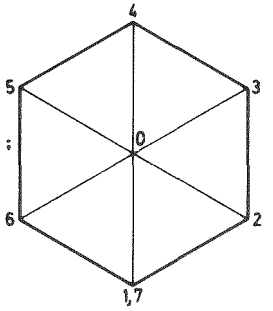
Externe Routinen sind im Hinblick auf einen modularen Aufbau und eine effektive Anwendung eines Systems von besonderer Bedeutung, da durch sie die Abspeicherung von Programmen in übersetzter Form für häufig auftretende Anwendungsfälle in Standardprogrammbibliotheken möglich ist. Diese Unterprogramme können dabei sowohl weitere Statements der bereits im Hauptprogramm aktivierten Sprache als auch Anweisungen eines neu definierten Subsystems enthalten. Abb. 3.14 demonstriert die Anwendung einer externen Routine zum gleichen Subsystem. Die Übersetzung des Unterprogrammes kann dabei mit Hilfe von PLS oder des Modulgenerators von REGENT erfolgen.

Dieser Modulgenerator, der auch die Erzeugung ausführbarer Moduln der Spracherweiterung unterstützt, kann benutzt werden, um dynamisch aufrufbare Programme mit Fähigkeiten anderer Subsysteme zu erstellen. Die hierzu erforderlichen Anweisungen lauten:

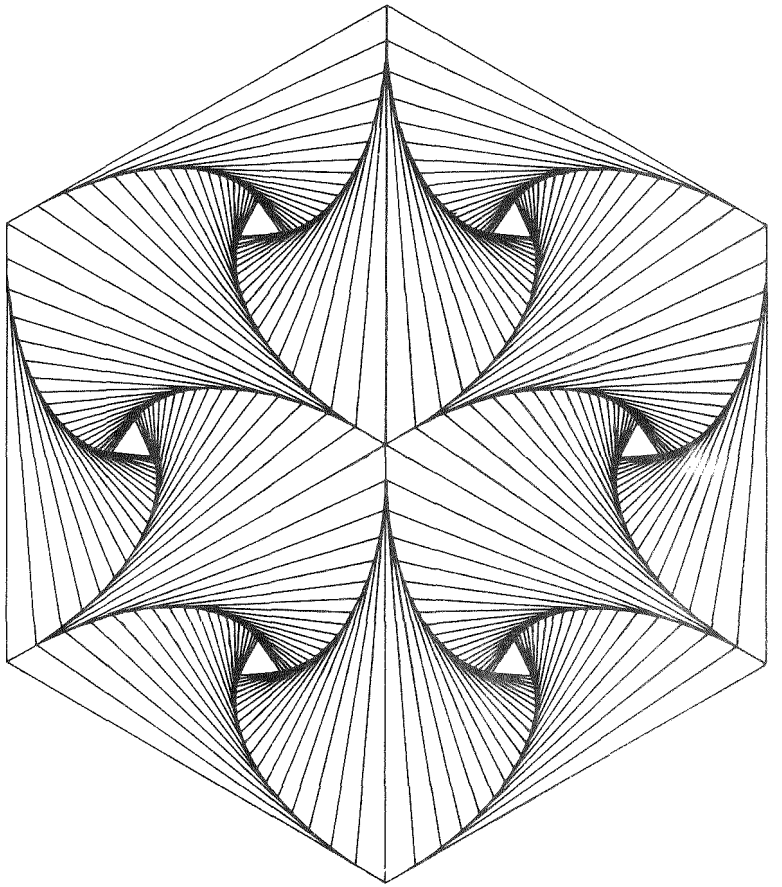
```
*SUBSYSTEM Y;
*MODUL  P;
*PROCESS REGENT;
 P:PROC(....)REGENT(SUB=Y);
   berechne: X,Y,P(X,Y)
   ENTER GIPSY;
     :
     DO  H=....;
         OPEN PLOT  L1 x L2;
         PLOT(NIVEAU(X,Y,P,H));
     END;
   END GIPSY;
 END P;
```

Das Beispiel verdeutlicht die Nutzung der graphischen Ausgabemöglichkeiten zur Darstellung von Wertefeldern eines Subsystems zur Behandlung von Problemen der Fluiddynamik. GIPSY stellt hierzu eine Funktion NIVEAU zur Ermittlung von Höhenlinien zum Werte H für P über X und Y (Abb. 3.15) und die Fähigkeiten zur RELIEF-Darstellung (Abb. 3.16) bereit.

```
/*********************************************************************/
TEST:PROC OPTIONS(MAIN) REGENT(NODA,NOMOD);
     ENTER GIPSY;
     DCL (P(4),SECHS(0:7)) POINT2;
     DCL ARTK ENTRY REGENT;
     SET SECHS(1)=SHIFT2(SECHS(0),0.,-100.);
     DO I=2 TO 7;
        SET SECHS(I)=ROTATE2(SECHS(I-1),60.);
     END;
     DO I=0 TO 7;
        SET SECHS(I)=SHIFT2(SECHS(I),110.,130.);
     END;
     OPEN PLOT DIN A4;
     DO I=2 TO 6 BY 2;
        DO J=-1,+1;
           SET P(1)=SECHS(0);
           SET P(2)=SECHS(I);
           SET P(3)=SECHS(I+J);
           CALL ARTK(P,20);
        END;
     END;
     END GIPSY;
  END TEST;
 /*********************************************************************/
```

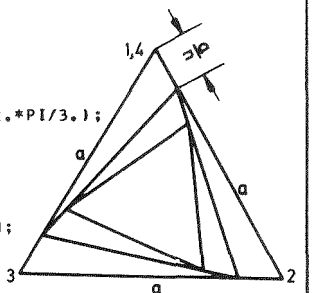


GIPSY - Hauptprogramm



GIPSY - Ausgabe

```
/*********************************************************************/
ARTK: PROC(P,I) REGENT(SUB=GIPSY);
     /*****************************/
     /* VERDREHUNG VON DREIECKEN  */
     /*****************************/
     DCL P(4) POINT2 PARAMETER;
     DCL (N,PI) DEC FLOAT(6);
     PI=3.1415E0;
     N=1.+(1./TAN(PI/(2.*I)))*SIN(2.*PI/3.)-COS(2.*PI/3.);
     DO J=1 TO I;
     SET P(4) = P(1);
     DO M=1 TO 3;
        PLOT(LINE(P(M),P(M+1)));
        POINT2_P(M).X(*)=POINT2_P(M).X(*)+
          (POINT2_P(M+1).X(*)-POINT2_P(M).X(*))/N;
     END;
     END;
END ARTK;
/*********************************************************************/
```



GIPSY - Unterprogramm

Abb. 3.14: Beispiel für die Anwendung einer externen Routine

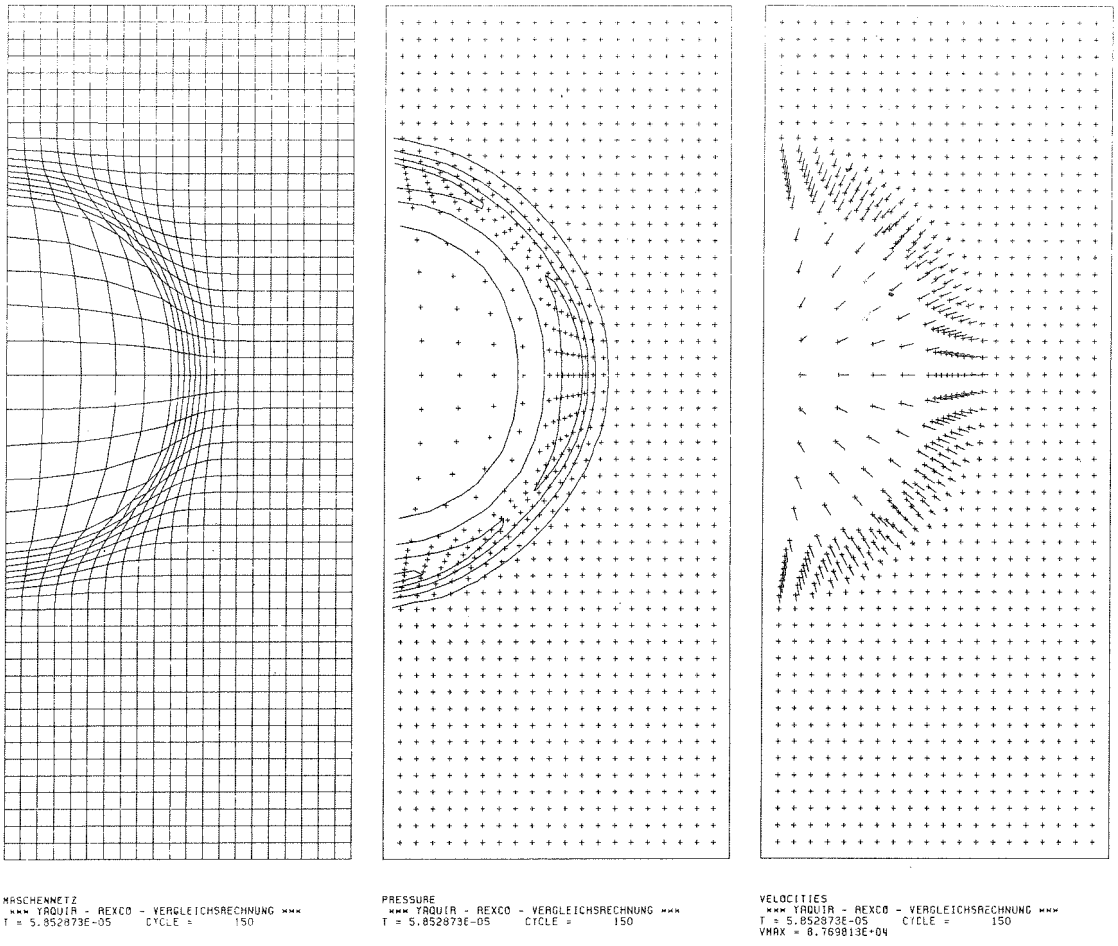


Abb. 3.15: Darstellung von Rechenergebnissen eines Fluiddynamik-Codes zu einem Explosionsvorgang (Mitte: Höhenlinienplot der Druckverteilung)

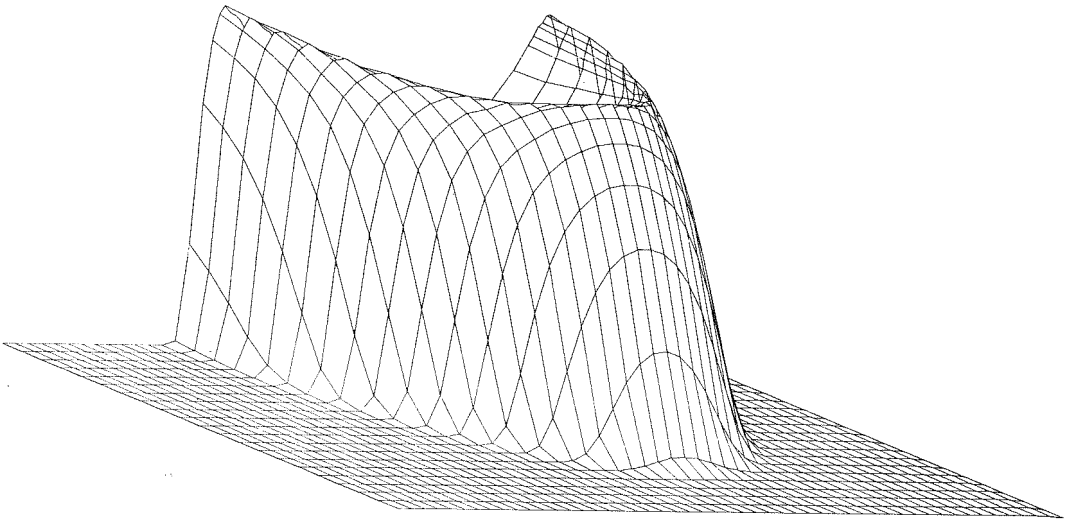


Abb. 3.16: Perspektivische Darstellung des Druckfeldes

4. Rechnerische Behandlung der Darstellungsprobleme räumlicher Objekte

Im letzten Abschnitt wurden die Fähigkeiten des GIPSY-Systems zur algorithmischen Behandlung graphischer Probleme behandelt, ohne auf diejenigen graphischen Datentypen näher einzugehen, die die Beschreibung räumlicher Objekte ermöglichen. Zwar tritt bereits bei den dort betrachteten dreidimensionalen graphischen Strichobjekten das Problem der Abbildung der Raumpunkte in eine zweidimensionale Bildebene und der Lineartransformation auf, doch stellen sie aufgrund ihrer maximal eindimensionalen Eigenausdehnung keine Forderungen nach Berücksichtigung von Durchdringungen oder Sichtbarkeitskriterien. Bei der zeichnerischen Darstellung von Körpern treten diese Probleme in den Vordergrund. In dieser Arbeit werden Lösungen für die effektive Behandlung der Körperkantenberechnung und der Sichtbarkeitsprüfung entwickelt.

Die Konzeption des in GIPSY realisierten Verfahrens wurde unter Abwägung der folgenden Gesichtspunkte erarbeitet:

- Flexibilität der Formulierungsmöglichkeiten für die Körperbeschreibung,
- Manipulierbarkeit und perspektivische Darstellung,
- Beschreibungsaufwand für Eingabe der Körperformen,
- Platzbedarf für die Objektspeicherung,
- Effektivität des Verfahrens zur Kantenberechnung und
- Anwendbarkeit eines effektiven Verfahrens zur Entscheidung der Sichtbarkeit.

Die Behandlung der räumlichen Objekte erfolgt im System unter Nutzung aller algorithmischen Fähigkeiten, die bereits im vorigen Abschnitt bei der Behandlung von Strichinformation vorgestellt wurde. Diese Eigenschaften werden im weiteren bei der Vorstellung der Raumelement- bzw. Körperbeschreibung noch verdeutlicht.

Eine weitere Voraussetzung für die Flexibilität bei der Problemformulierung ist die Vielfalt der Funktionen zur Erzeugung der graphischen Objekte sowie die uneingeschränkte Anwendungsmöglichkeit der Transformationen auf die räumlichen Objekte (vgl. Abb. 3.8). In keinem der bisher bekannten Systeme werden diese Fähigkeiten bereitgestellt.

Die mathematischen Grundlagen dieser Berechnungen werden im nächsten Abschnitt dargelegt.

4.1 Lineartransformationen und perspektivische Abbildung

In der Technik wird in der Regel eine Darstellung von Werkstücken durch mehrere orthogonale Parallelprojektionen vorgenommen, für deren Interpretation in DIN-Normen festgelegte Vorschriften die Grundlage bilden. Einer mit dieser Darstellungsart vertrauten Person ist eine räumliche Rekonstruktion des Objektes materiell oder auch nur in der geistigen Vorstellung möglich. Eine Assoziation des Räumlichen wird durch Verwendung von Zentralprojektionen erleichtert, die jedoch bei der manuellen zeichnerischen Darstellung einen ungleich höheren Aufwand erfordern.

Die Zeichnungserstellung beruht auch heute noch weitgehend auf den konstruktiven Methoden der klassischen Darstellenden Geometrie /104,138/. Bei den Systemen zur Behandlung und Darstellung von räumlichen Objekten mit dem Rechner ist eine rechnerische Bestimmung von aus einer Projektion resultierenden Bildpunkten unumgänglich und bei der Rechengeschwindigkeit der modernen Datenverarbeitungsanlagen auch bei großen Punktemengen durchführbar. Die Eigenschaften der hierzu herangezogenen Verfahren variieren stark.

In der Mehrzahl der graphischen Systeme sind jedoch nur Parallel- (oder Zentral-) Projektionen auf eine fest vorgegebene Bildebene möglich. In der Standard-4x4-Matrix /41,42/, die diese Abbildung übernimmt, sind auch die Angaben für eine räumliche Lineartransformation enthalten. Aus dieser Implementierung folgt, daß keine unabhängige Transformation von Objekten durchführbar ist. Die Abbildungsrechnung jedes Punktes ist zudem mit den Operationen zur Lineartransformation belastet.

Diese eingeschränkten Fähigkeiten von Projektionsalgorithmen sind in einigen Fällen jedoch auch Voraussetzung für die Realisierung spezieller Verfahren zur Umrißkantenbestimmung /88/ oder Sichtbarkeitsentscheidung (Prioritätenverfahren).

Im vorliegenden System GIPSY sind die Lineartransformationen und die Abbildungsfunktion in voneinander unabhängigen Operationen realisiert. Für die Lineartransformationen gelten die in der Mathematik allgemein üblichen Beziehungen, die in Anhang C zusammengestellt sind. Ihre Anwendung in der Sprache erfolgt nach den in Abschnitt 3.3.3 und Anhang A angegebenen Regeln.

Zur Abbildung der räumlichen Strukturen stellt GIPSY einen Projektionsalgorithmus bereit, der keinen Einschränkungen hinsichtlich Lage der Bildebene oder Wahl der Projektionsrichtung bei Parallel- und Zentralprojektion unterliegt. Auch bei der Behandlung der Sichtbarkeit von Körperkanten gilt diese Allgemeinheit. Die Herleitung des Abbildungsgesetzes, das auf von Grimm /63/ entwickelten Methoden basiert, wird im Anhang D vorgenommen.

Aus den obengenannten Eigenschaften ergeben sich folgende Vorteile bei der Anwendung der Transformations- und Abbildungsalgorithmen in GIPSY:

- Die Ergebnisse von Lineartransformationen, wie Translation, Skalierung und Rotation, können bei der Formulierung der Probleme den graphischen Daten zugewiesen und für eine weitere Manipulation abgerufen werden, wodurch unabhängige Transformationen von einzelnen Teilen eines Bildes möglich sind.
- Gleiches gilt für die Ergebnisse der Ausführung einer Abbildungsrechnung, die neben ihrer Anwendung bei der direkten graphischen Ausgabe, z.B. auch die Reduktion eines Raumpunktes oder räumlichen Polygonzuges auf eine fiktive Bildebene ermöglicht und entsprechenden zweidimensionalen Objekten zuweist.
- Diese Abbildungen sind als beliebige Parallel- oder Zentralprojektion durchführbar.
- Die Bildebene kann bei allen Projektionen - solange sich dabei ein Schnittpunkt mit den Projektionsstrahlen ergibt - frei gewählt werden, wodurch sich z.B. eine Möglichkeit zum 'Zoomen' bietet.
- Diese Eigenschaften schränken nicht die Möglichkeiten des verwendeten Visibilitätskriteriums ein.
- Alle diese Transformationen und Projektionen beeinflussenden Parameter können dynamisch während der Programmausführung geändert werden.

4.2 Beschreibung von Körpern und Darstellung durch in ihrer Oberfläche befindliche Kanten

4.2.1 Aufwand zur Eingabebeschreibung von räumlichen Objekten

Jeder Darstellung eines räumlichen Objektes geht dessen Beschreibung in einer maschinenverständlichen Form voraus. Von der Flexibilität und vom Aufwand dieser Beschreibungsmöglichkeiten hängt in starkem Maße die Einsatzfähigkeit eines Systems ab. Die Eingabe für nicht regelmäßige räumliche Objekte muß durch den Aufbau der Körperform mittels eines Vorrates an verfügbaren Grundelementen erfolgen, die sich entsprechend ihrer Eigenausdehnung in die Hierarchie:

- Punkt (0),
- Linie (1),
- Fläche (2) und
- Raum (3)

einordnen lassen. Es ist zu erwarten, daß der Beschreibungsaufwand umso geringer wird, je kleiner die Differenz zwischen der Dimension des räumlichen Objektes und der der beschreibenden Raumelemente - bei gleichzeitiger guter Gestaltübereinstimmung - ist.

Es gibt zahlreiche Versuche,den Beschreibungsaufwand durch die Ausnutzung häufig auftretender Werkstückformen zu minimieren. Das Schwergewicht dieser Arbeiten von Balogh /15/, Butz /17/, Kurth /80/, Lacoste /86/, Schnelle /132/ und Simon /141/ liegt jedoch nicht auf der Erzeugung einer dreidimensionalen Werkstückrepräsentation im Rechner, sondern bei Algorithmen zur direkten zweidimensionalen zeichnerischen Ausgabe der fertigungsgerechten Unterlagen. Dabei sind diese Beschreibungssysteme in hohem Maße auf die Ausnutzung spezieller - vorwiegend rotationssymmetrischer - Eigenschaften der Werkstücke angewiesen. Eine perspektivische Abbildung von Körpern ist dabei nicht möglich. Die räumliche "Darstellung" übernimmt dort eine auf der Basis dieser Beschreibungen programmierte Werkzeugmaschine, die durch die Ausführung der Arbeitsschritte gewissermaßen die Fähigkeiten eines räumlichen Displaygerätes realisiert.

Dieser Abbildungsvorgang muß in graphischen Systemen durch komplizierte Algorithmen ersetzt werden, denen diese häufig an Bearbeitungsgängen orientierte Eingabeform entgegensteht oder nicht ausreichend Information liefert. Dennoch ist auch bei denjenigen Systemen, die mehr von graphischer Seite das Problem angehen, eine Tendenz zur Beschreibung mit Grundelementen höherer Eigenausdehnung (≥ 2) zu erkennen /22,80,124/.

Anfänglich wurden in der graphischen Datenverarbeitung Körper (vorwiegend Polyeder) durch die Eingabeliste von Punkten, Kanten und Flächen beschrieben. Durch die Orientierung der dabei entstehenden flächenumschließenden Polygonzüge wurde bei einigen Verfahren auf die Innen- bzw. Außenseite des Körpers geschlossen.

Sieht man von den graphischen Flächen, die aufgrund ihrer Unregelmäßigkeit nur durch eine Vielzahl von Punkten in dieser Fläche bestimmt werden können, ab, so ist für die räumlichen Objekte eine Eingabe durch Flächen bzw. Volumina anzustreben. Der Vorteil, der sich daraus ergibt, soll am Beispiel regelmäßiger Polyeder deutlich gemacht werden.

Polyeder	F	E	K	$\frac{F}{E+K}$
Tetraeder	4	4	6	0.4
Hexaeder	6	8	12	0.3
Oktaeder	8	6	12	0.44
Dodekaeder	12	20	30	0.24
Ikosaeder	20	12	30	0.47

Euler'scher Polyedersatz:

$F + E - K = 2$

(F = Fläche,
K = Kante,
E = Ecke)

Abb. 4.1: Eingabeaufwand für Polyeder

Dieser einfache Vergleich macht bereits die deutliche Reduzierung des Eingabeumfangs ($< 50\%$) bei der Beschreibung durch Flächen deutlich. Eine ausführliche Diskussion des Beschreibungsaufwandes wird von Kurth /80/ vorgenommen, wobei eine weitere Zusammenfassung der Flächen zu sogenannten Formelementen gefordert wird.

Bei der Erweiterung der Systemfähigkeiten auf analytisch beschreibbare Flächen ergibt sich zwangsläufig eine Festlegung der Körperform durch Flächen

und deren Parameter. Woon /164/ und Wenz /142/ verwenden diese Eingabeformen in ihren Systemen, während Braid /22/ einfache räumliche Grundformen anbietet. In allen Systemen wird mit der Eingabe der Flächen bzw. Volumina eine Aussage über die angenommene Verteilung der Materie an diesen Flächen vorgenommen.

Im vorliegenden System GIPSY wird ebenfalls diese effektive Methode zur Vereinbarung von Körpern genutzt. Die zur Beschreibung verfügbaren Flächenformen sind jedoch dabei erstmals als Datentypen in einer algorithmischen Sprache realisiert. Die Möglichkeiten, die sich daraus - auch für die Gruppierung von Flächen zu abrufbaren Definitionen von häufig auftretenden räumlichen Grundformen - ergeben, werden später noch diskutiert werden. An dieser Stelle sollten nur die Gründe für die Auswahl von Flächen zur Objektbeschreibung dargelegt werden.

4.2.2 Aufwand zur Darstellung von räumlichen Objekten

Der Darstellungsaufwand für die körperbegrenzenden Flächen wird von der rechnerinternen und -externen Repräsentation der Daten bestimmt (vgl. Abschnitt 2.2). Für eine effektive Beschreibung von Objekten scheidet die aus einer internen Pflasteraufteilung gewonnene Darstellung durch Rasterlinien aus. Dieser Aufwand ist nur gerechtfertigt bei Flächen, die sich einer analytischen Behandlung entziehen, zumal trotz des großen Aufwandes dieses Verfahren durch die verwendeten Approximationsfunktionen innerhalb der Pflaster nur eine näherungsweise Beschreibung der Flächen erlaubt.

Beeinträchtigend kommt hinzu, daß die zeichnerische Ausgabe einer Vielzahl von Rasterlinien nicht zur Verdeutlichung der Abbildung beiträgt und daß sich die Behandlung einer etwaigen Durchdringung mehrerer Flächen nach dieser Methode sehr schwierig gestaltet.

Im vorliegenden System GIPSY wird deshalb eine Liniendarstellung von Körpern mittels

- Durchdringungs- und
- betrachtungsabhängigen Umrißkanten

realisiert, die auf dem geometrisch unabdingbaren Minimum von Angaben zur Eingabe und internen Speicherung basiert.

Diese Repräsentation der Objekte bietet folgende Vorteile:

- geringer Eingabeaufwand,
- geringer Speicheraufwand,
- Genauigkeitssteigerung der Kurvendarstellung verursacht lediglich linearen Anstieg des Rechenaufwand ohne größeren Speicheraufwand,
- zeichnerische Ausgabe entspricht der menschlichen Wahrnehmung und
- bei Lineartransformationen ist nur eine geringe Anzahl flächenbestimmender Daten umzurechnen.

Die auf dieser Basis entwickelte analytische Methode zur Berechnung von Durchdringungs- und Umrißkurven wird im folgenden dargelegt.

4.2.3 Auswahl der möglichen Flächenformen und deren mathematische Behandlung

Neben den bisher behandelten DV-technischen Gesichtspunkten ist für die Auswahl der Flächenformen vor allem die Wahrscheinlichkeit ihres Auftretens bei technischen Objekten und damit deren Anwendungspotential von Bedeutung. Die Oberflächen von Werkstücken sind - insbesondere nach spangebender Bearbeitung - häufig durch eine Vielzahl von Teilflächen ebener, kugeliger, zylindrischer oder kegeliger Gestalt bestimmt. Simon /141/ bestätigt diese Annahme durch statistische Erhebung, nach denen etwa 85% der betrachteten Bauteile durch eine Kombination dieser geometrischen Grundformen darstellbar sind.

Die Kugel-, Zylinder- und Kegelflächen sind damit die für technische Anwendungen interessante Untermege der Flächen 2.Ordnung. Durch die Hinzunahme der Ebene wird die Flächenauswahl vervollständigt. Der Aufwand zur Darstellung derartig begrenzter räumlicher Objekte wird durch die Möglichkeiten zur exakten internen Beschreibung und die Reduzierung der Berechnung auf die bei normaler Betrachtung wahrgenommenen Linienzüge der Durchdringungs- und Umrißkurven verringert.

Die Torusfläche und ihre Durchdringungen, die im technischen Bereich z.B. bei Rohrkrümmern oder Ausrundungsradien auftreten, kann, da es sich dabei um Schnitte mit einer Fläche 4. Ordnung handelt, mit den im weiteren beschriebenen analytischen Verfahren nicht beherrscht werden.

Alle in Abschnitt 2.3 verglichenen bisherigen Realisierungen nehmen eine derartige Behandlung vor, verwenden dazu aber ausschließlich numerische Methoden, die neben einem höheren Aufwand in vielen Fällen Einschränkungen hinsichtlich Manipulierbarkeit, Projektionsart, Kantenbestimmung oder Sichtbarkeitsalgorithmus bedingen.

Die Mathematik hat sich ausgiebig mit den Flächen 2. Ordnung /31,135/ und den bei ihrem Schnitt entstehenden Raumkurven 4. Ordnung 1. Spezies /136,18/ befaßt. Die Quadriken werden dort Bewegungen bzw. Koordinatentransformationen in ihre Hauptachsen unterzogen und dann in einer normierten Form ohne gemischtquadratische Glieder klassifiziert und z.B. auf Symmetrieeigenschaften hin untersucht /61,108/. In der Literatur

läßt sich aber kein Ansatz zur Lösung des Durchdringungsproblems zweier Quadriken in der für technische Anwendungen wichtigen allgemeinen Lage finden. Lediglich für sogenannte cyklische Kurven, d.h. den Schnitt einer Kugel mit einer Fläche 2. Ordnung und hier insbesondere für das Viviani-Fenster (Schnitt: Kugel-Zylinder in ausgezeichneter Anordnung), sind Ergebnisse vorhanden.

Generell sind für die Lösung des Durchdringungs- bzw. Umrißproblems folgende Wege begehbar:

- numerisches Aufsuchen gemeinsamer Punkte der am Schnitt beteiligten Quadriken sowie von Punkten, die durch das Anlegen von Tangentialebenen entstehen,
- Schneiden der Flächen 2. Ordnung mit einer Schar von Hilfsebenen und anschließendes numerisches Bestimmen gemeinsamer Punkte der entstehenden ebenen Kegelschnitte,
- Herleitung eines allgemeinen analytischen Ansatzes und
- Berechnung spezieller analytischer Lösungsfunktionen für die Schnittkombinationen und Umrisse der verschiedenen Flächen.

Mit Ausnahme des Systems von Becker /20/ fallen alle bisherigen Realisationen /88,167,164,142(,22)/ in die erste Kategorie dieser Aufteilung (vgl. Abb. 2.5). Von den analytischen Lösungen ist der im vorliegenden System beschrittene spezielle Weg vorzuziehen, da er durch optimale Anpassung an die Charakteristiken einer Schnittpaarung die effektivste Bestimmung von Kurvenpunkten ermöglicht.

Zur Beschreibung von Quadriken wird allgemein die quadratische Form

$$a_{11}x_1^2 + a_{22}x_2^2 + a_{33}x_3^2 + 2a_{12}x_1x_2 + 2a_{13}x_1x_3 + 2a_{23}x_2x_3$$

$$+ 2a_{o1}x_1 + 2a_{o2}x_2 + 2a_{o3}x_3 + a_{oo} = 0,$$

die sich aus der Matrixschreibweise in inhomogenen Koordinaten

$$\underline{x}^T \underline{\underline{A}}\, \underline{x} + 2\underline{a}^T \underline{x} + a_{oo} = 0$$

herleiten läßt, herangezogen. Da basierend auf diesem Ansatz nur eine numerische Lösung möglich ist, wird bei der im folgenden behandelten Berechnung von einer vektoriellen Beschreibung der Flächen ausgegangen.

Der Ansatz für die Durchdringung ermöglicht durch die Verwendung einer rein vektoriellen

$$Q\ (\underline{x}) = 0$$

und einer parametrischen Darstellung

$$\underline{x} = \underline{x}\ (u,v)$$

der Flächen Ebene, Kugel, Zylinder und Kegel die Ermittlung einer Beziehung

$$v = v\ (u),$$

wodurch gleichzeitig eine explizite parametrische Darstellung der Schnittkurve

$$\underline{x}_s = \underline{x}\ (u)$$

gegeben ist.

Bevor auf die Berechnung der Schnittkurven der verschiedenen Flächenkombinationen eingegangen wird, sollen die hierzu erforderlichen Gleichungen in der obengenannten rein vektoriellen und parametrischen Form zusammengestellt werden. Zur Veranschaulichung der geometrischen Beziehungen in den folgenden Gleichungen dient Abb. 4.2.

Gleichungsformen

Die Gleichungen der Ebene sind /61/:

$$Q_E\ (\underline{x}_E)=(\underline{x}_E - \underline{p}_E)\ \underline{n}_E = 0 \qquad (1)$$

und

$$\underline{x}_E(u,v) = \underline{p}_E + u\cdot\underline{a} + v\cdot\underline{b}\ . \qquad (2)$$

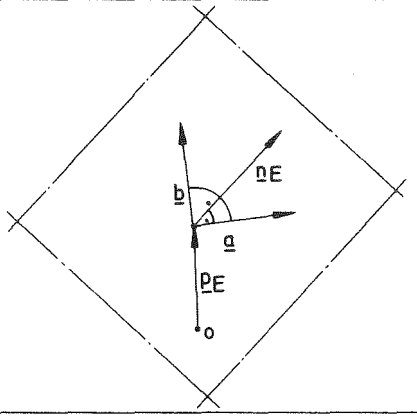
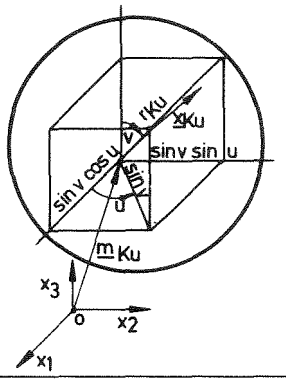
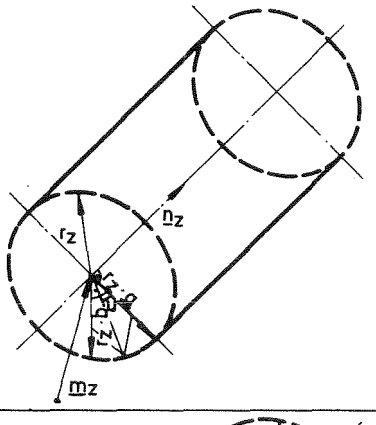
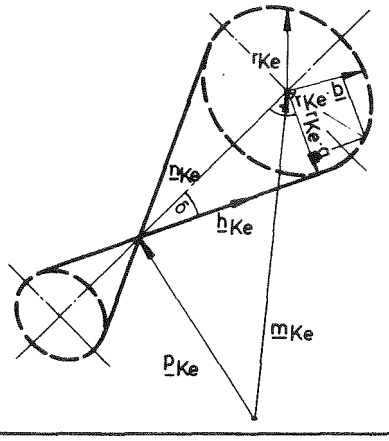
FLÄCHE	DARSTELLUNG	GLEICHUNG (vektoriell)	GLEICHUNG (parametrisch)
EBENE		$(\underline{X} - \underline{p}_E)\,\underline{n}_E = 0$	$\underline{X} = \underline{p}_E + u\,\underline{a} + v\,\underline{b}$ $(\underline{a}, \underline{b}) \perp \underline{n}_E$
KUGEL		$(\underline{X} - \underline{m}_{Ku})^2 = r_{Ku}^2$	$\underline{X} = \underline{m}_{Ku} + r_{Ku}\,(\underline{a} \times \underline{b})$ $(\underline{a} = (-\sin u, \cos u, 0),$ $\underline{b} = (-\cos u \cos v, -\sin u \cos v, \sin v))$ $\underline{X} = \underline{m}_{Ku} + r_{Ku}\,(\cos u \sin v\,\underline{e}_1 + \sin u \sin v\,\underline{e}_2 + \cos v\,\underline{e}_3)$
ZYLINDER		$r_Z = \dfrac{\lvert(\underline{X} - \underline{m}_Z) \times \underline{n}_z\rvert}{\lvert\underline{n}_Z\rvert}$	$\underline{X} = \underline{m}_Z + r_z\,\underline{a}\cos u + r_Z\,\underline{b}\sin u + v\,\underline{n}_Z$ $(\underline{a}, \underline{b}) \perp \underline{n}_Z$
KEGEL		$(\underline{m}_{Ke} - \underline{p}_{Ke})(\underline{X} - \underline{p}_{Ke}) =$ $\lvert\underline{m}_{Ke} - \underline{p}_{Ke}\rvert\lvert\underline{X} - \underline{p}_{Ke}\rvert\cos\delta$	$\underline{X} = \underline{p}_{Ke} + v\,[(\underline{m}_{Ke} - \underline{p}_{Ke}) + r_{Ke}\,\underline{a}\cos u + r_{Ke}\,\underline{b}\sin u]$ $(\underline{a}, \underline{b}) \perp \underline{n}_{Ke}$

Abb. 4.2: Bestimmungsgrößen der Flächen Ebene, Kugel, Zylinder und Kegel

In vektorieller Form ist eine Kugel bestimmt durch

$$Q_{Ku}(\underline{x}_{Ku}) = (\underline{x}_{Ku} - \underline{m}_{Ku})^2 - r_{Ku}^2 = 0 \; . \tag{3}$$

Parametrisch gilt:

$$\underline{x}_{Ku}(u,v) = \underline{m}_{Ku} + r_{Ku}(\cos u \sin v \, \underline{e}_1 + \sin u \sin v \, \underline{e}_2 + \cos v \, \underline{e}_3) \tag{4}$$

Die vektorielle Form der Gleichung eines Zylinders wird der Beziehung zur Errechnung des Abstandes eines Punktes von einer Geraden entnommen /77/, so daß

$$Q_z(\underline{x}_z) = \frac{|(\underline{x}_z - \underline{m}_z) \times \underline{n}_z|}{|\underline{n}_z|} - r_z = 0 \; . \tag{5}$$

Die parametrische Beziehung der Zylindergleichung geht von einem achsennormalen Kreis /63/:

$$\underline{x}_K = \underline{m}_z + r_z (\underline{a} \cos u + \underline{b} \sin u)$$

aus, bei dem $\underline{a}$ und $\underline{b}$ zwei orthonormierte Vektoren in der Ebene senkrecht zur Achse des Zylinders mit Richtung $\underline{n}_z$ darstellen. Diese Vektoren werden nach dem im Anhang E beschriebenen E.Schmidt'schen Orthogonalisierungsverfahren bestimmt.

Senkrecht zu diesem Basiskreis und damit parallel zur Zylinderachse läuft auf einer Mantellinie der Parameter v, wodurch

$$\underline{x}_z(u,v) = \underline{m}_z + r_z(\underline{a} \cos u + \underline{b} \sin u) + v \cdot \underline{n}_z \tag{6}$$

ist.
Durch die Parameterbereiche $0 \leq u < 2\pi$ und $-\infty < v < \infty$ ist damit ein unendlich ausgedehnter Kreiszylinder beschrieben.

Die vektorielle Behandlung eines Kegels ist durch die Anwendung des Skalarproduktes zwischen der Mittellinie mit einer Mantellinie möglich:

$$\underline{n}_{Ke} \, \underline{h}_{Ke} = |\underline{n}_{Ke}| \; |\underline{h}_{Ke}| \cos \delta$$

Die Vektoren $\underline{n}_{Ke}$ und $\underline{h}_{Ke}$ in dieser Beziehung können durch Differenzen ersetzt werden, so daß gilt:

$$Q_{Ke}(\underline{x}_{Ke}) = (\underline{m}_{Ke} - \underline{p}_{Ke})(\underline{x}_{Ke} - \underline{p}_{Ke}) - |\underline{m}_{Ke} - \underline{p}_{Ke}||\underline{x}_{Ke} - \underline{p}_{Ke}| \cos \delta \qquad (7)$$

Parametrisch kann ein Punkt $\underline{x}$ auf der Kegelhülle angegeben werden durch:

$$\underline{x}_{Ke} = \underline{p}_{Ke} + v \cdot \underline{h}_{Ke}$$

mit

$$\underline{h}_{Ke} = \underline{m}_{Ke} + r_{Ke}(\underline{a} \cos u + \underline{b} \sin u) - \underline{p}_{Ke}.$$

Hieraus ergibt sich wiederum eine Gleichungsform, die einen Basiskreis, aufgespannt mit den Vektoren $\underline{a}$ und $\underline{b}$ senkrecht zur Rotationsachse, beinhaltet:

$$\underline{x}_{Ke}(u,v) = \underline{p}_{Ke} + v \left[(\underline{m}_{Ke} - \underline{p}_{Ke}) + r_{Ke}(\underline{a} \cos u + \underline{b} \sin u) \right] \qquad (8)$$

Die Parameterbereiche von u und v zur Beschreibung eines unendlich ausgedehnten Kegels entsprechen denen des Zylinders.

Für die Behandlung und Speicherung der obengenannten Flächenformen stellt die GIPSY-Sprache entsprechende Datentypen bereit, die nach den Regeln aus Abschnitt 3.3 deklariert werden. Diese Objekte sind intern repräsentiert durch die geometrisch notwendigen Angaben.
Im einzelnen werden benötigt:

- Ebene: $\underline{p}_E$, $\underline{n}_E$,
- Kugel: $\underline{m}_{Ku}$, r_{Ku},
- Zylinder: $\underline{m}_z$, $\underline{n}_z$, r_z und
- Kegel: $\underline{p}_{Ke}$, $\underline{m}_{Ke}$, r_{Ke}.

Zur Erzeugung dieser Flächendaten können verschiedenartige Eingabeformen genutzt werden und zu ihrer Manipulation stehen Lineartransformationen bereit (vgl. Abb. 3.8 und Anhang A). Auf die Beschreibung von räumlichen Objekten durch diese Flächen und die bei einer automatischen Analyse von Körpern auftretenden Probleme wird in Abschnitt 4.5 eingegangen.

4.2.4 Eigenschaften der Durchdringungskurven

Die Berechnung der Durchdringungskurven auf der Basis der Gleichungen (1) bis (8) wird im Anhang F1 wiedergegeben. Neben den mehr anschaulichen Überlegungen der Bestimmung der Schnitte des Flächenkataloges mit Ebenen und von Kugeln untereinander, sind die Herleitungen der komplizierten Schnittverläufe mit Hilfe der Vektoralgebra auszugsweise wiedergegeben. Nach zahlreichen Umformungen und Vereinfachungen unter Verwendung der Lagrange Identität

$$(\underline{a}x\underline{b})(\underline{c}x\underline{d}) = (\underline{a}\underline{c})(\underline{b}\underline{d}) - (\underline{b}\underline{c})(\underline{a}\underline{d})$$

ergeben sich die in Abb. 4.3 wiedergegebenen Beziehungen. Die dort verzeichneten sieben Gleichungstypen lassen sich durch Vernachlässigung von Gliedern der trigonometrischen Formen spaltenweise aus dem jeweils komplexesten Schnittverlauf für eine Flächenart auf nur vier verschiedenartige Gleichungen reduzieren. Durch diese vier auf der Diagonalen stehenden Beziehungen lassen sich alle zwischen Ebenen, Kugeln, Zylindern und Kegeln auftretenden Durchdringungen in beliebiger Lage berechnen.

Für die rechnerische Behandlung sind folgende Eigenschaften von Bedeutung:

- die Programmierung dieser Kurvenverläufe ist einfach und in der Ausführung effektiv,
- die trigonometrischen Formen enthalten neben Sinus- und Cosinus-Funktionen nur Konstanten, die vor der inkrementalen Berechnung der Kurve ermittelt werden können,
- die Kurven durchlaufen einen definierten Parameterbereich $0 \leq u < 2\pi$ (auf die Ausnahme des geraden Schnittverlaufes wird später noch eingegangen), dessen Inkremente zur Parametersteuerung variierbar sind,
- die komplizierten Durchdringungsformen bilden, bedingt durch die Wurzeln, zwei Kurvenäste,
- diese Kurvenäste beinhalten jedoch untereinander zusammenhängende Punktefolgen und
- über die Existenz der Kurve kann durch das Vorzeichen des Radikanten entschieden werden.

	EBENE	KUGEL	ZYLINDER	KEGEL
EBENE	$\underline{D} + \underline{A}\ u$	$\underline{\text{cosin}}$	$\underline{\text{cosin}}$	$\underline{D} + \dfrac{\underline{\text{cosin}}}{\text{cosin}}$
KUGEL	–	$\underline{\text{cosin}}$	$\underline{\text{cosin}} + \underline{D}\ (D \pm \sqrt{\text{cosin}})$	$\underline{D} + \underline{\text{cosin}}\ \dfrac{\text{cosin} \pm \sqrt{\text{cosin2}}}{D}$
ZYLINDER	–	–	$\underline{\text{cosin}} + \underline{D}\ (\text{cosin} \pm \sqrt{\text{cosin2}})$	$\underline{D} + \underline{\text{cosin}}\ \dfrac{\text{cosin} \pm \sqrt{\text{cosin2}}}{\text{cosin2}^*}$
KEGEL	–	–	–	$\underline{D} + \underline{\text{cosin}}\ \dfrac{\text{cosin} \pm \sqrt{\text{cosin2}}}{\text{cosin2}^*}$

$$\underline{\text{cosin}} = \underline{C} + \underline{A} \cos u + \underline{B} \sin u$$

$$\text{cosin} = C + A \cos u + B \sin u$$

$$\text{cosin2} = D + E \cos u + F \sin u + G \cos 2u + H \sin 2u$$

Abb. 4.3: Gleichungsformen der Schnitt- und Umrißkurven

Diese Eigenschaften der berechneten Kurvenverläufe erlauben eine Programmierung der Algorithmen zur Durchdringungsberechnung, die hinsichtlich Speicherplatzbedraf und Rechenzeit effektiv ist.

Die Anwendung des analytischen Verfahrens ist auch für den Einsatz von Kleinrechnung zur Behandlung der Darstellungsprobleme komplizierter Körper geeignet.

4.2.5 Eigenschaften der Umrißkurven

Der wahre Umriß einer Fläche ist gegeben durch diejenigen Punkte der Fläche, die beim Anlegen von Tangenten durch das Betrachtungszentrum als Berührungspunkte entstehen. Die Berechnung des Umrisses einer Fläche 2. Ordnung läßt sich auf den Schnitt dieser Flächen mit einer durch das aktuelle Betrachtungszentrum (Pol) bestimmten sogenannten Polarebene zurückführen. Verfahren dieser Art /142/ behandeln die Umrißkurvenberechnung dann formal wie eine Durchdringung, ohne dabei Vorteile aus den einfachen Verläufen der Umrisse zu ziehen.

Die wahren Umrisse der Flächen Kugel, Zylinder und Kegel werden hier wie die Durchdringungen aus analytischen Berechnungen ermittelt.

In Anhang F2 sind die Gleichungen für den als Kreis auftretenden wahren Umriß einer Kugel und die jeweils zwei ausgezeichneten Mantellinien von Zylinder und Kegel für eine explizite Berechnung hergeleitet. Diese Umrißverläufe entsprechen den Gleichungen

$$\underline{x} = \underline{\text{cosin}} \quad \text{und} \quad \underline{x} = \underline{D} + \underline{A} \cdot u$$

in Abb. 4.3 für die Durchdringungen. Nach der Bestimmung der Konstanten in diesen Gleichungen reduziert sich die Berechnung auf das Durchlaufen der Parameterintervalle und ist dadurch wesentlich effektiver als nur das numerische Aufsuchen von gemeinsamen Punkten von Quadrik und Polarebene.

Die Fähigkeiten der Methoden zur Kantenberechnung werden in Abb. 4.4 an einem Komplexteil demonstriert. Die Darstellung enthält die Umriß- und Durchdringungskurven für alle im System GIPSY verfügbaren Flächenformen.

Im folgenden soll auf die Methode zur Beschreibung räumlicher Objekte und auf die automatische Analyse der dabei erzeugten internen Struktur eingegangen werden.

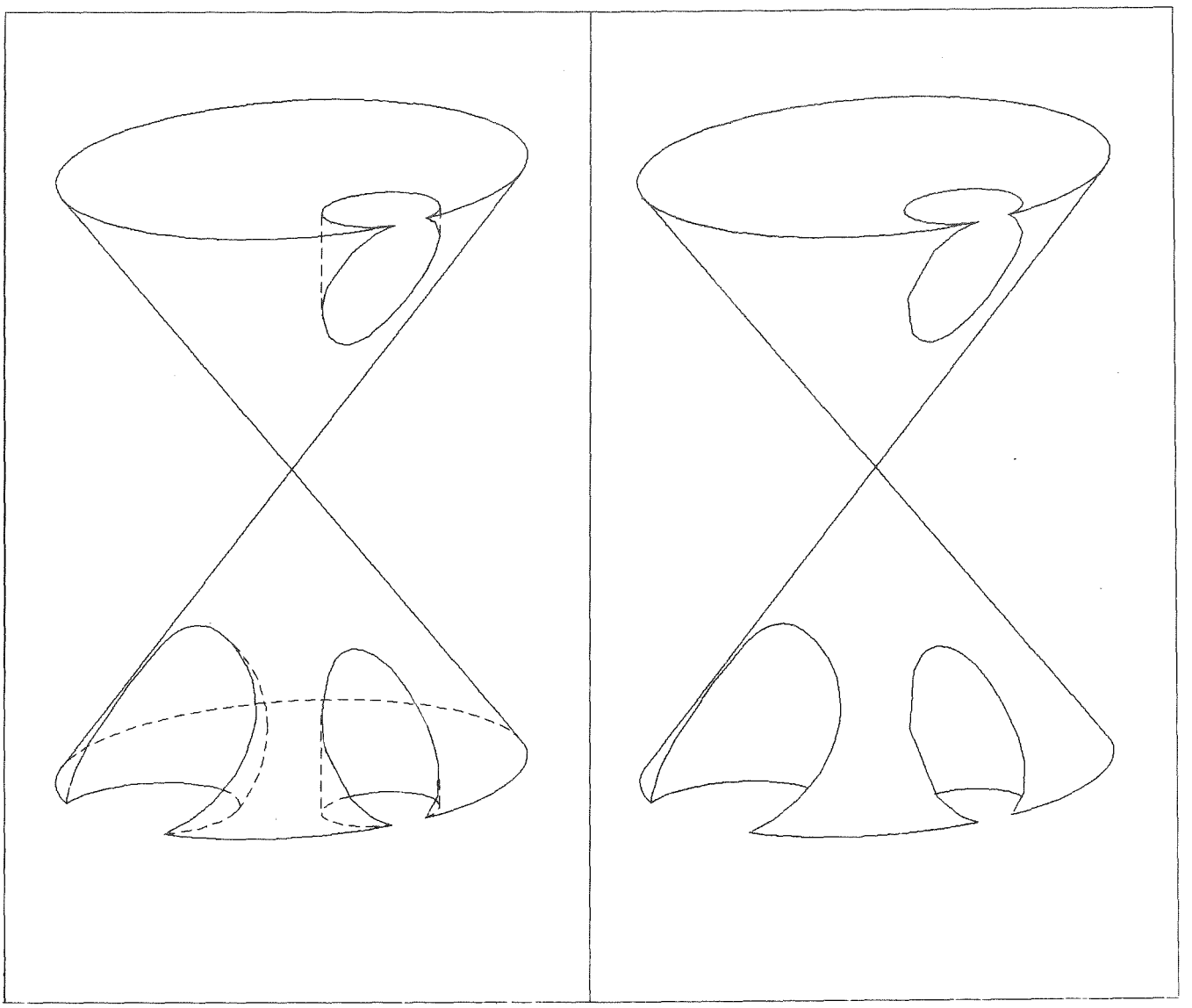


Abb. 4.4: Demonstrationskörper für Schnitt- und Umrißkanten

4.3 Raumdefinition zu Flächen

Die oben gefundenen expliziten Beziehungen der Durchdringungs- und Umrißkanten sind für eine rein mathematische Behandlung der Flächen ausreichend. Bei einer Beschreibung von räumlichen Objekten im technischen Sinne ist es erforderlich, die entstehenden Raumkurven mit einer Aussage über die Existenz dieser Kurven bezüglich eines zu definierenden Körpervolumens zu verknüpfen.

Diese Angabe ist durch zwei Anforderungen an die zeichnerische Darstellung bedingt:

- die Unterdrückung unsichtbarer Kanten oder Kantenstücke soll - sofern gewünscht - erfolgen und
- die Wiedergabe soll auf die auf der Körperoberfläche befindlichen Kanten beschränkt bleiben.

Bei den Verfahren zur Behandlung von Polyedern, die ihre Körperbeschreibung durch die Eingabe der Hierarchie von Punkten, Kanten und Flächen erfahren - dies gilt fast für alle -, ist nur die erste Forderung relevant, da die zweite durch die Eingabeform automatisch erfüllt ist.

In den Arbeiten über Sichtbarkeitsprobleme an Polyedern wird der Aspekt der körperlichen Repräsentanz der Objekte deshalb selten ausdrücklich diskutiert, obwohl alle Verfahren auf der Nutzung dieser Eigenschaft über die Orientierung der einschließenden Ebenen beruhen.

Das Verfahren von Roberts /122/ beschreibt für seinen Punktetest einen Körper durch konvexe Teilkörper, deren Raumpunkte durch ein lineares Gleichungssystem festgelegt sind. Appels /1/ 'quantitative invisibility' separiert sogenannte Konturkanten aus dem Schnitt von front- und rückseitigen Ebenen und Loutrel /90/ scheidet unsichtbare von potentiell sichtbaren Kanten bei seiner 'edge classification' durch die Bestimmung der Winkel zwischen der Betrachtungsrichtung und der Flächenanordnung. Beide Kantentests nutzen dabei die Orientierung der Ebenen.

Die Richtung der Normalen weist dabei ins Innere der Volumina, d.h. sie drückt aus, auf welcher Seite der Fläche sich 'Material' befindet. Für die mathematische Behandlung des Problems reicht die unterschiedliche Kennzeichnung des Zustandes des Raumes beiderseits der Fläche aus. Die Anschauung wird jedoch durch die der Realität entsprechende Zuordnung von Materie erleichtert.

Bei der Beschreibung von Körpern, die durch Quadriken umschlossen sind, wird keine Angabe über den Verlauf von Durchdringungskanten auf der Oberfläche gemacht, wie dies bei Polyedern geschieht. Die aus der Angabe einer Flächenpaarung resultierende Schnittkurve muß gegenüber anderen Flächen auf Zugehörigkeit zum Körpervolumen untersucht werden.

Ähnlich wie bei Ebenen wird bei Woon /164/ und Wenz /142/ den Flächen 2. Ordnung eine 'polarity' bzw. ein Materialbereich zugeordnet. Braid /22/ baut seine Körper direkt aus einfachen räumlichen Grundelementen auf. Das System von Becker /20/ kennt jedoch nur Flächen in ihrer mathematischen Bedeutung. Zur Begrenzung der Flächenausdehnung finden zusätzlich - ähnlich wie bei Weiss /167/ - Endbegrenzungen als einschließendes Polyeder Anwendung.

Im vorliegenden System wird ausschließlich von der mathematischen Erstrekkung und den darauf basierenden mengentheoretischen Definitionen ausgegangen. Zu diesem Zweck wird eine Funktion vereinbart, die eine Zuordnung zwischen einer Fläche und dem zugehörigen Definitionsbereich durch eine logische Aussage herstellt:

$$\text{IS\_MAT}\ (\underline{x}, S_i) = \left\{ \begin{array}{ll} 1 & \text{im Definitonsbereich} \\ 0 & \text{außerhalb des Definitionsbereiches} \end{array} \right\} \text{der Fläche } S_i$$

Für einen Raumpunkt $\underline{x}$ gelten bezüglich der Flächen Ebene, Kugel, Zylinder und Kegel die Bedingungen nach Abb. 4.5.

FLÄCHE	BEDINGUNG FÜR IS_MAT='1'B
EBENE	$(\underline{X} - \underline{p}_E) \underline{n}_E \geqslant 0$
KUGEL	$\lvert \underline{X} - \underline{m}_{Ku} \rvert \leqslant r_{Ku}$
ZYLINDER	$\lvert (\underline{X} - \underline{m}_Z) \times \underline{n}_Z \rvert \leqslant r_Z$
KEGEL	$\frac{\lvert (\underline{X} - \underline{p}_{Ke}) \times (\underline{m}_{Ke} - \underline{p}_{Ke}) \rvert}{\lvert \underline{m}_{Ke} - \underline{p}_{Ke} \rvert} \leqslant \frac{(\underline{X} - \underline{m}_{Ke}) (\underline{m}_{Ke} - \underline{p}_{Ke}) r_{Ke}}{\lvert \underline{m}_{Ke} - \underline{p}_{Ke} \rvert^3}$

Abb. 4.5: Bedingung für Zugehörigkeit eines Punktes zum Definitionsbereich einer Fläche

Durch diese Beziehungen ist umgekehrt der Definitionsraum einer Fläche S_i bestimmt durch:

$$D_{S_i} = \left\{ \underline{x} \mid \text{IS\_MAT} (\underline{x}, S_i) \right\}$$

4.4 Operationen mit Flächen zur Bildung von Raumelementen

Unter Heranziehung der obigen Definition lassen sich abgeschlossene, endliche Volumina, sogenannte Raumelemente, durch die Bildung der Schnittmenge der Raumpunkte der Definitionsbereiche aller Hüllflächen bilden. Für diese Aussage gilt formal:

$$D_{Sp} = (D_{S_1} \cap D_{S_2} \cap \ldots \cap D_{S_n}) = \bigcap_{i=1}^{n} D_{S_i} \quad ,$$

wobei für alle beteiligten Flächenkombinationen die Beziehung

$$D_{S_i} \cap D_{S_j} \neq \emptyset$$

erfüllt sein muß, um ein endliches, Material erfülltes Volumen zu erhalten.

Zur Formulierung der obengenannten Schnittmenge bietet GIPSY die Möglichkeit, an der Mengenbildung beteiligte Flächen durch den '+'-Operator miteinander zu verknüpfen, sofern sie mit Normalorientierung verwendet werden soll. Eine Umkehrung des Definitionsbereichs einer Fläche kann durch die Angabe des '-'-Operators, der sogenannten Inversion, erreicht werden. Durch eine einfache Anwendung beim Schnitt zweier Kugeln soll die Bedeutung dieser Operationen verdeutlicht werden:

Deklaration zweier Kugeln und eines Raumelementes für zwei Flächen.

```
DCL  (G,K)  BALL,
     REST   SPACE(2);
```

Die beteiligten Flächen lassen sich durch die obigen Operatoren zu folgenden Raumelementen verbinden:

K \ G	↓	↑
→	G+K (1)	-G+K (3)
←	G-K (2)	~~-G-K (4)~~

(1) SET REST = SPACE(G+K);

Die letzte Operation (4) führt nicht zu einem endlichen Volumen und scheidet deshalb aus. Die Ergebnisse der obigen Anweisungen (1-3) sind spaltenweise von unten nach oben in Abb. 4.6 wiedergegeben, wobei die Lage der kleineren Kugel von links nach rechts variiert.

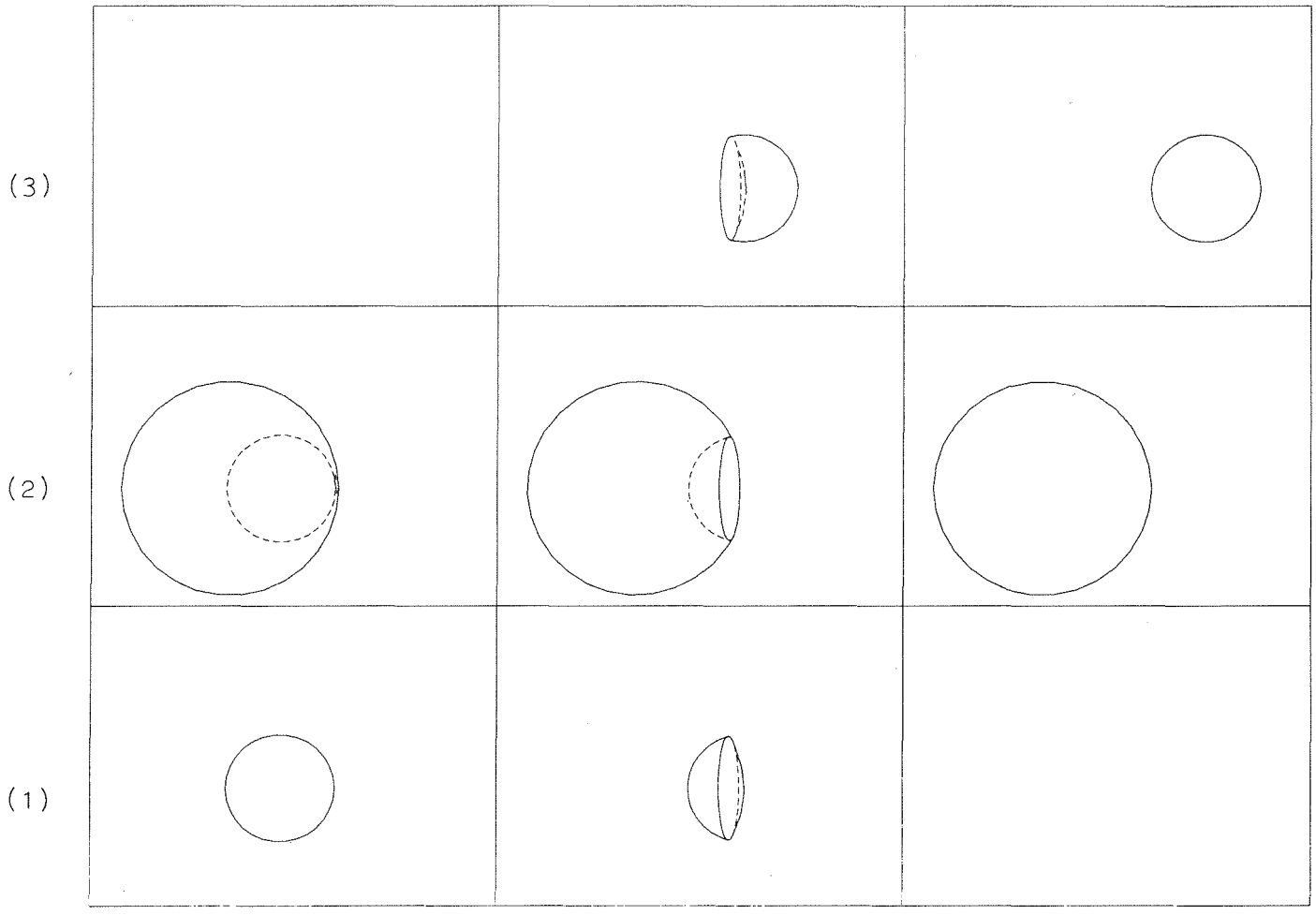


Abb. 4.6: Spezifikation eines Raumelementes aus zwei Kugeln

Bei der Beschreibung der Raumelemente stehen wie bei allen anderen graphischen Objekten algorithmische Fähigkeiten zur Verfügung. In Abb. 4.7 werden diese zur Formulierung eines Programmes für einen Dodekaeder und eine Pyramide genutzt.

Die Festlegung der Flächenanordnung erfolgt dabei durch das Zusammenwirken von Schleifenprogrammierung und graphischen Operationen in effektiver Weise:

```
DO  I=1 TO 12;
    SET DF(I)=PLANE(DP(I),DP(0));
    SET DODEKAEDER=SPACE(DODEKAEDER+DF(I));
END;
```

```
SAMPLE:PROC OPTIONS(MAIN) REGENT(PLOT=STATOS,NOCA);
ENTER GIPSY;
 DCL DP(0:12) POINT;
 DCL DF(12) PLANE;
 DCL DODEKAEDER SPACE(12);
 DCL PYRAMIDE SPACE(6);
  SET DP(1)=SHIFT(DP(C),0.,C.,100.) ;
  SET DP(2)=SHIFT(DP(0),0.,0.,-100.) ;
  DO I=3 TO 4;
     SET DP(I)=ROTATE(DP(1),0.,(I-2)*60.,C.) ;
  END;
  SET DP(3)=ROTATE(DP(3),-36.,C.,0.) ;
  DO I=3 TO 9 BY 2;
     SET DP(I+2)=ROTATE(DP(I),72.,0.,0.) ;
     SET DP(I+3)=ROTATE(DP(I+1),72.,C.,C.) ;
  END;
  DO I=1 TO 12;
     SET DF(I)=PLANE(DP(I),DP(0)) ;
     SET DODEKAEDER=SPACE(DODEKAEDER+DF(I));
  END;
  SET PYRAMIDE=SPACE(PLANE(DP(0),DP(1)));
  DO I=3 TO 11 BY 2;
     SET PYRAMIDE=SPACE(PYRAMIDE+DF(I));
  END;
  CHANGE PROJECTION PARALLEL -1.,-0.2,-0.1,
         INCREMENT LINEAR 20.,
         PROJECTION ORIGIN 160.,60.;
  OPEN PLOT DIN A3;
  PLOT(PYRAMIDE);
  CHANGE PROJECTION ORIGIN 160.,160.;
  OPEN PLOT DIN A3;
  PLOT(DODEKAEDER);
END GIPSY;
FINISH;
END SAMPLE;
```

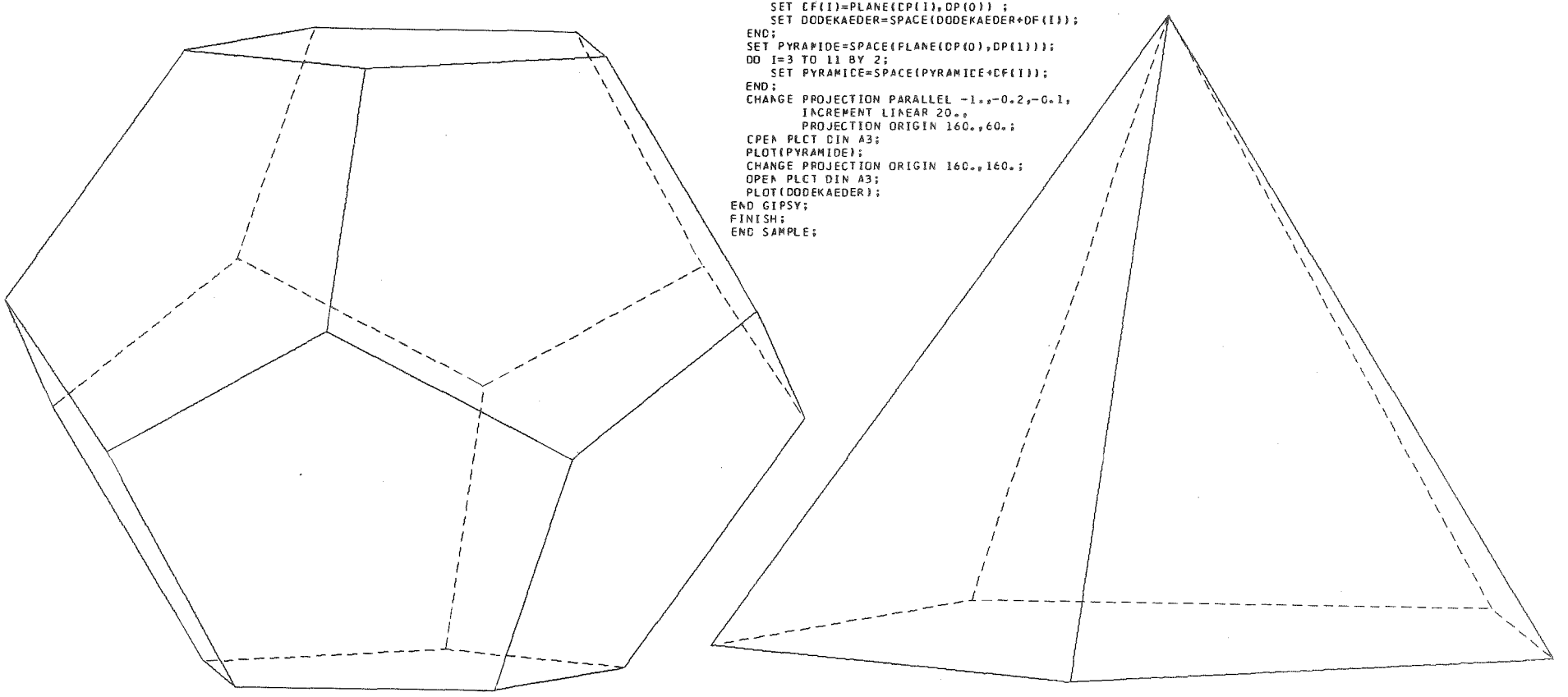


Abb. 4.7: Algorithmische Beschreibung von Polyedern

Durch die Verwendung einer prozeduralen Sprache lassen sich aus einer einmal erfolgten Strukturdefinition durch Parametervariation vielfältige Formen gewinnen. Abb. 4.8 und Abb. 4.9 liefern einen anschaulichen Beweis für die Mächtigkeit dieser Beschreibungstechnik, in dem sie das vollständige Programm seiner graphischen Ausgabe gegenüberstellen. In diesem Beispiel wird die Orientierung einer Hüllfläche invertiert. Neben dieser dynamischen Umkehrung des Definitionsbereiches beim Einbau einer Fläche in ein Raumelement,kann eine statische auf die Fläche bezogene Änderung durch die Operation SWITCH vorgenommen werden (vgl. Anhang A).

```
SAMPLE:  PROC OPTIONS(MAIN) REGENT(PLOT,NODA);
ENTER    GIPSY;
  DCL    WUERFEL SPACE(6),
         KUGEL BALL,
         PKT(0:1) POINT,
         (X(3,3) INIT(100.,(2)((3)0.,100.)),R) DEC FLOAT(6);
  DO   I=1 TO 3;
       SET PKT(1)=POINT(X(I,1),X(I,2),X(I,3));
       SET WUERFEL=SPACE(WUERFEL+PLANE(PKT(0),PKT(1))+PLANE(PKT(1),PKT(0)));
  END;
  CHANGE  PROJ CENTRAL 800.,-100.,800.,
          PROJ ORIGIN   70.,70.;
  DO   R=45. TO 75. BY 10.;
       IF R>60. THEN CHANGE PROJ PARA -80.,10.,-18.;
       OPEN PLOT DIN A4 BROAD;
       PLOT(SPACE(WUERFEL+KUGEL));
       OPEN PLOT DIN A4 BROAD;
       PLOT(SPACE(WUERFEL-KUGEL));
  END;
END GIPSY;
FINISH;
END SAMPLE;
```

Abb. 4.8: Programmbeispiel eines Raumelementes

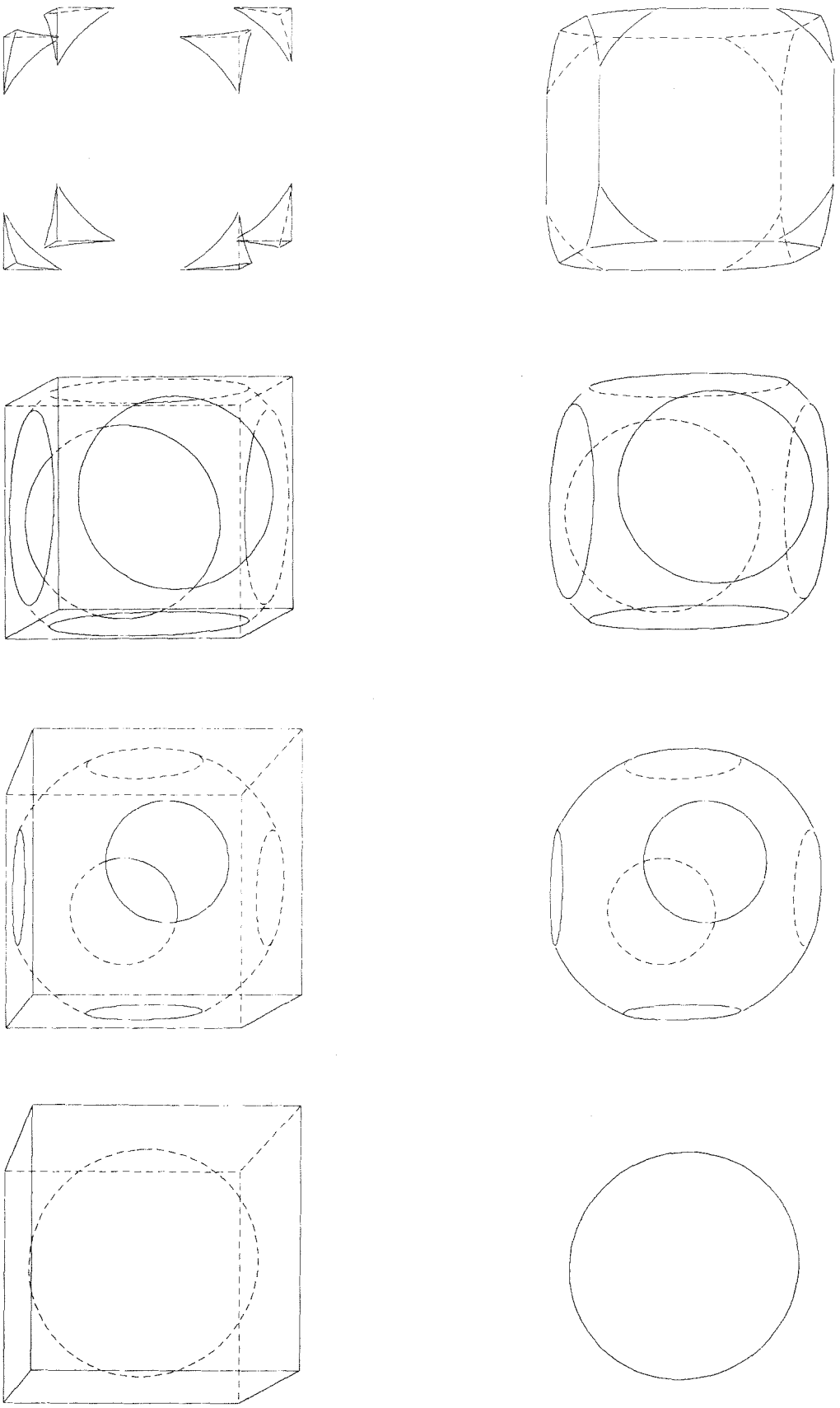


Abb. 4.9: Ausgabe zu Abb. 4.8

4.5 Behandlung von Körpern

Durch die bisher angebotene Möglichkeit zur Vereinbarung von Raumelementen sind viele Körper nicht behandelbar, da die unendliche Ausdehnung der Flächen Ebene, Zylinder und Kegel - entsprechend ihrer mathematischen Bedeutung (vgl.Abschnitt 4.2.3) - und die damit vorgenommene Raumdefinition (vgl. Abschnitt 4.3) die Anordnung von Körpervolumina beiderseits einer Begrenzungsfläche verbietet. Bezogen auf Polyeder bedeutete dies, daß nur konvexe Objekte /101/ beschreibbar sind. Bei der versetzten Anordnung zweier Quader gemäß Abb. 4.10, die als ein Körper betrachtet werden sollen, bedingt die für Raumelemente anzuwendende Bildung der Schnittmenge der Hüllebene des einen Quaders mit denen des anderen die teilweise bzw. im Falle der Berührungsfläche die vollständige Auslöschung des Körpervolumens.

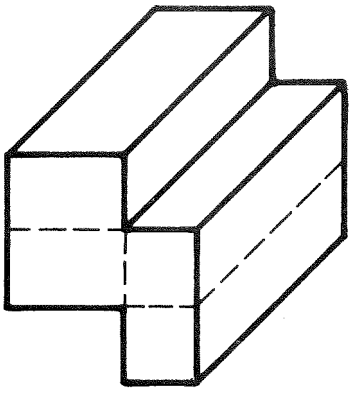


Abb. 4.10: Konvexes Polyeder

Dieser Konflikt kann beseitigt werden, wenn dieses Objekt nicht als ein Raumelement, sondern als eine Zusammenfassung von Raumelementen betrachtet wird, wobei die bei dieser Zusammenfügung auftretenden Probleme - wie das Erkennen des Wegfalls von Kanten oder Stücken davon in der Berührungsfläche - bei der automatischen Körperanalyse gelöst werden müssen.

Im vorliegenden System wird diese Zusammenfassung durch das Einbringen von Raumelementen in eine Kollektion realisiert. Mathematisch steht hinter dieser Operation die Bildung der Vereinigungsmenge der Volumina der am Körperaufbau beteiligten Raumelemente:

$$D_{CO} = (D_{SP_1} \cup D_{SP_2} \cup \ldots \cup D_{SP_n}) = \bigcup_{i=1}^{n} D_{SP_i}$$

Aus der Zusatzbedingung

$$D_{SP_i} \cap D_{SP_j} = \emptyset \quad \text{für } i \neq j$$

ergibt sich, daß keine gegenseitige Durchdringung der einen Körper definierenden Raumelemente auftreten darf. Die Volumina der Raumelemente eines Körpers müssen nicht miteinander in Verbindung stehen. Tun sie es dennoch, so ist die Berührungsfläche eine sogenannte virtuelle Fläche, die über einen Namen identifiziert, aber durch Inversion mit unterschiedlicher Orientierung in einem der beteiligten Raumelemente versehen wird.

Auf die aus dieser Körperdefinition resultierenden Eigenschaften der Algorithmen zur Körperanalyse wird später noch eingegangen werden. Im nächsten Abschnitt soll zunächst auf die Möglichkeiten zur Formulierung von Anweisungen zur Körperdarstellung und die dabei verwendeten Datenstrukturen Bezug genommen werden.

4.5.1 Beschreibung des Körperaufbaus und resultierende Datenstruktur

In Abb. 4.11 ist im rechten Bildteil ein Körper dargestellt, der sich durch das Zusammenfügen der beiden linken Raumelemente ergibt. Die Aufteilung dieses Objektes in zwei Raumelemente ist erforderlich, da sich das Körpervolumen auf beide Seiten einer begrenzenden Fläche - der Bohrung - verteilt.

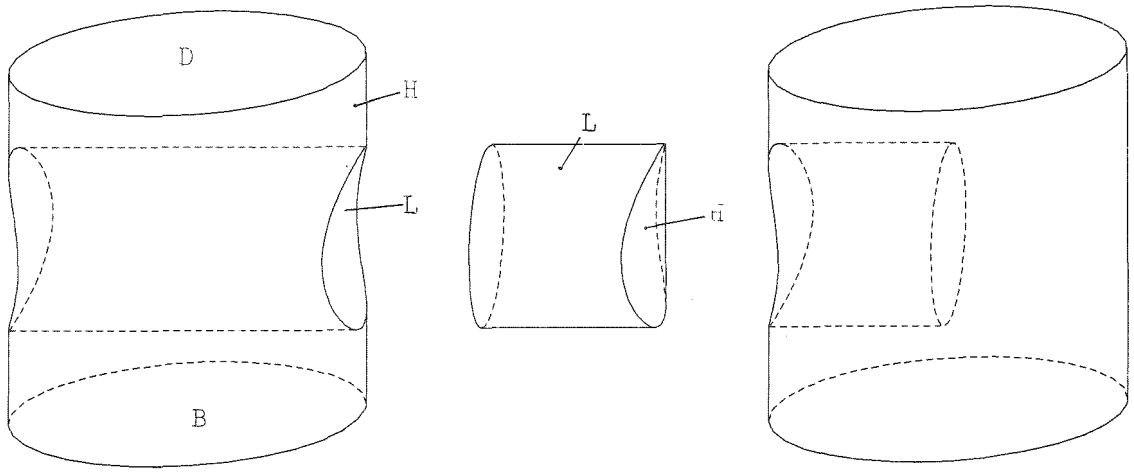


Abb. 4.11: Körperaufbau

Zur Beschreibung der Oberflächen werden Datentypen der Spracherweiterung benutzt:

```
DCL  (D,B)  PLANE,
     (H,L)  CYLINDER;
```

Nach erfolgter Zuweisung geometrischer Attribute sollen diese Variablen zur Definition der Teilvolumina

```
DCL  TEIL(2)  SPACE(4);
```

mittels

```
SET  TEIL(1) = SPACE(H-L+D+B);
```

und

```
SET  TEIL(2) = SPACE(L+H+PLANE(...));
```

beitragen, die ihrerseits einen Körper durch

```
DO  I=1 TO 2;
    SET OBJECT=COLL(OBJECT+TEIL(I));
END;
```

spezifizieren.

Die Relationen, die durch die obigen Anweisungen zwischen den Objekten aufgebaut werden, lassen sich der schematischen Darstellung der Datenstruktur in Abb. 4.12 entnehmen. Jede Referenz (⊕) repräsentiert eine durch Verweisinformation hergestellte Abhängigkeit von Objekten.

Im Kernspeicher finden diese Beziehungen ihren Niederschlag in einer Datenstruktur, deren Realisation in Abb. 4.13 verdeutlicht werden soll. Auf die deklarierten graphischen Objekte (D,B,H,L) und das Ergebnis einer PLANE-Operation (W) wird durch Zeiger aus den SPACE-Objekten verwiesen, neben denen jeweils eine Angabe über die Orientierung und Sichtbarkeit einer Fläche steht. Die Körperanalyse benutzt diese Angaben, um über die aus Schnitt- und Umrißberechnung folgernden Kurvenverläufe zu entscheiden.

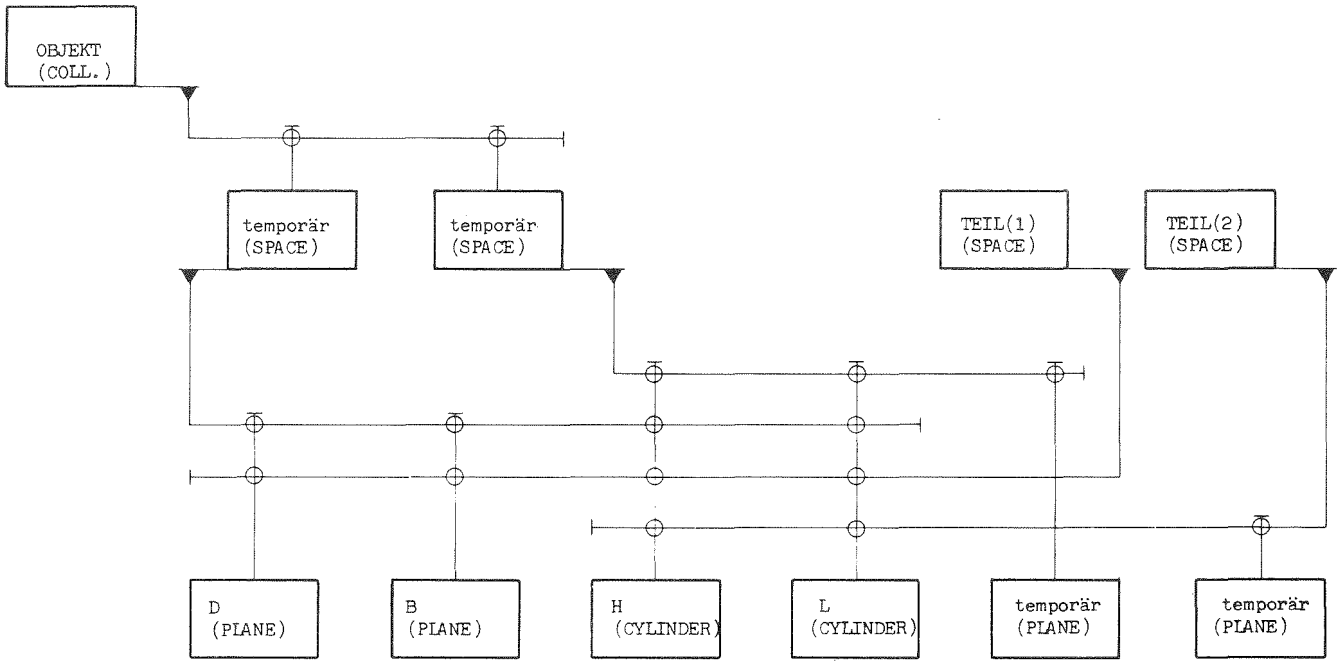


Abb. 4.12: Schematische Darstellung der Datenstruktur

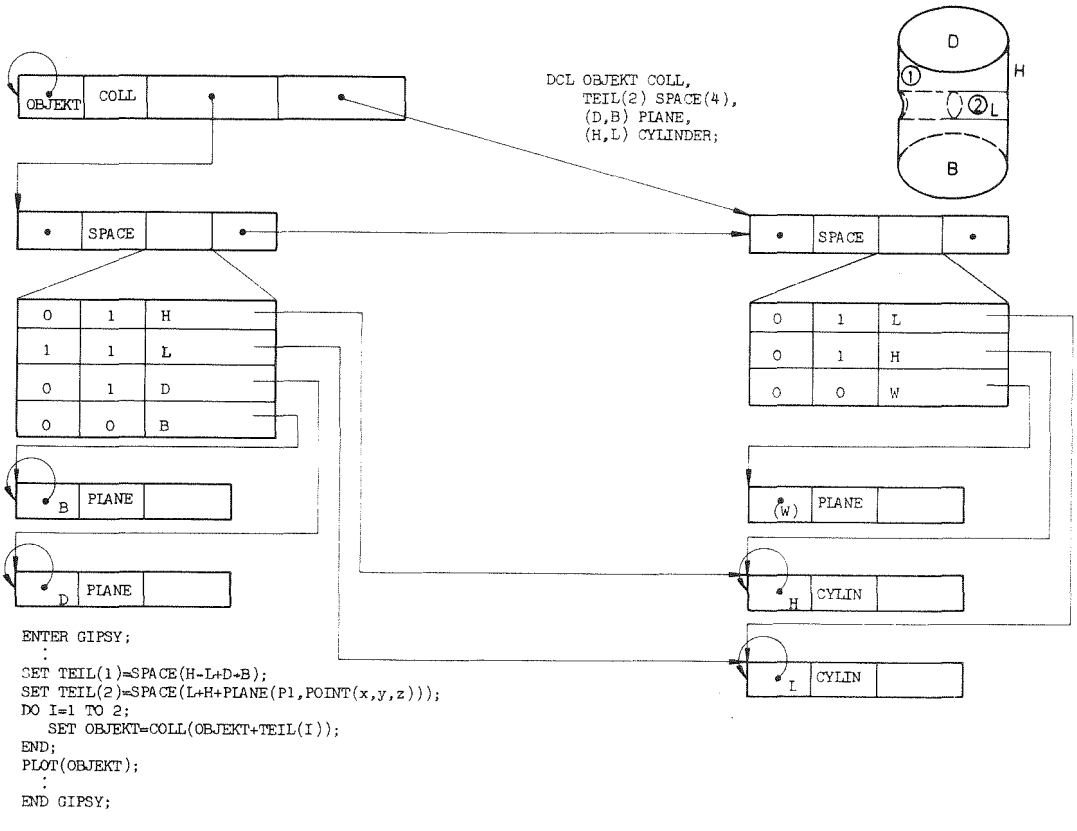


Abb. 4.13: Realisierung der Datenstruktur

4.5.2 Ablauf der Analyse der Struktur räumlicher Objekte

Die Analyse der einen Körper repräsentierenden Struktur orientiert sich an der Hierarchie:

- Körper (COLLECTION),
- Raumelement (SPACE),
- Fläche (PLANE,BALL,CYLINDER,CONE) und
- Kurve.

Sie hat die Aufgabe, eine automatische Darstellung der Körper mit ihren

- Durchdringungs- und
- Umrißkurven

zu ermöglichen. Hierzu werden die durch den Schnitt oder Umriß von Flächen entstehenden Raumkurven unter Einbeziehung der topologischen Angaben aus den Raumelementen und Körpern berechnet. Die Kurve selbst entsteht dabei durch die Interpretation von in Algorithmen gespeicherten Erzeugungsvorschriften, die auf den Angaben der geometrischen Kenngrößen der beteiligten Flächen basierend jeden möglichen Kurvenverlauf beinhalten.

Raumelementanalyse

Die Erklärung der automatischen Analyse der Struktur soll zunächst anhand eines Beispieles zur Ausgabe eines Raumelementes - hier TEIL(1) aus Abb. 4.13 - vorgenommen werden. Eine vollständige Ermittlung der zur Darstellung notwendigen Körperkanten liegt dann vor, wenn alle möglichen Schnitte von Flächenpaarungen und Umrisse zu allen Flächen eines Raumelementes ermittelt wurden. In Abb. 4.14 sind die aus der Anwendung dieser Aussage auf TEIL(1) resultierenden Beziehungen angegeben.

Kontur	Schnitt	H	L	D	B
x	H		v	x	x
v	L			¬	¬
	D				¬
	B				

¬ kein Schnitt

x Schnitt bzw. Umriß berechnen

v virtuelle Fläche

Abb. 4.14: Schnittpaarungen und Umrisse eines Raumelementes

Die Untersuchung eines Raumelementes (vgl. Abb. 4.15) gliedert sich dabei in die Berechnung von Schnitt- und Umrißkurven. Aus Symmetriegründen fallen Schnitte zwischen zwei Flächen nur einmal an, so daß sich die Kombinationen auf diejenigen auf einer Seite der Diagonalen in obiger Matrix reduzieren. Durch dieses Vorgehen mit Bildung von Zweierkombinationen ergibt sich eine quadratische Abhängigkeit zwischen der Anzahl der beschreibenden Flächen und dem Aufwand zur Kantenberechnung. Wird eine Schnittpunktbestimmung aus Dreierkombinationen von Flächen vorgenommen, so ergibt sich ein kubischer Zusammenhang /164,142/. Die verbleibenden Flächenkombinationen werden vor einer Ausführung der Schnittoperation auf die prinzipielle Möglichkeit des gegenseitigen Durchdringens (IS_POS) untersucht. Fällt diese Prüfung negativ ('¬') aus, so werden sie bei der Kantenberechnung übergangen. Im vorliegenden Fall verbleiben die Schnitte H-L, H-D und H-B, wobei die Paarung H-L wegen der Kennzeichnung 'virtuell' des Schnittpartners L erst später näher betrachtet werden soll.

Für die Flächenkombinationen H-D und H-B können aus der Gleichung (f3) in Anhang F1 die charakterisierenden Konstanten der resultierenden Schnittkurve gewonnen werden.

Für diesen Kurventyp (Schnitt: Ebene-Zylinder, vgl. Abb. 4.3) sind damit die Vorbereitungen für die explizite Berechnung des Kurvenverlaufes, der sich über einen festen Parameterbereich ($0 \leq u < 2\pi$) erstreckt, abgeschlossen.

Entsteht bei der Schnitt- oder Umrißbestimmung eine Gerade, so muß deren Erstreckung durch eine zusätzliche Untersuchung der Raumelement- bzw. Körperausdehnung ermittelt werden (RANGE). Dabei werden die Schnittpunkte dieser Geraden mit allen Flächen des Raumelementes nach den Beziehungen aus Anhang F3 bestimmt und der Parameterbereich auf $u_B < u < u_E$ eingeschränkt.

Die Bestimmung der Konstanten für die Umrißkurven von Zylindern erfolgt nach den in Anhang F2 hergeleiteten Beziehungen, wobei die zu diesem Zeitpunkt gültigen Projektionsbedingungen in die Berechnung eingehen. Da es sich bei den Konturlinien von Zylindern um Geraden handelt, muß auch hier eine Festlegung des Parameterbereichs vorgenommen werden, bevor die eigentliche Kantenberechnung gestartet wird. Der Einfluß der Virtualität auf Umrißkurven soll ebenfalls später untersucht werden.

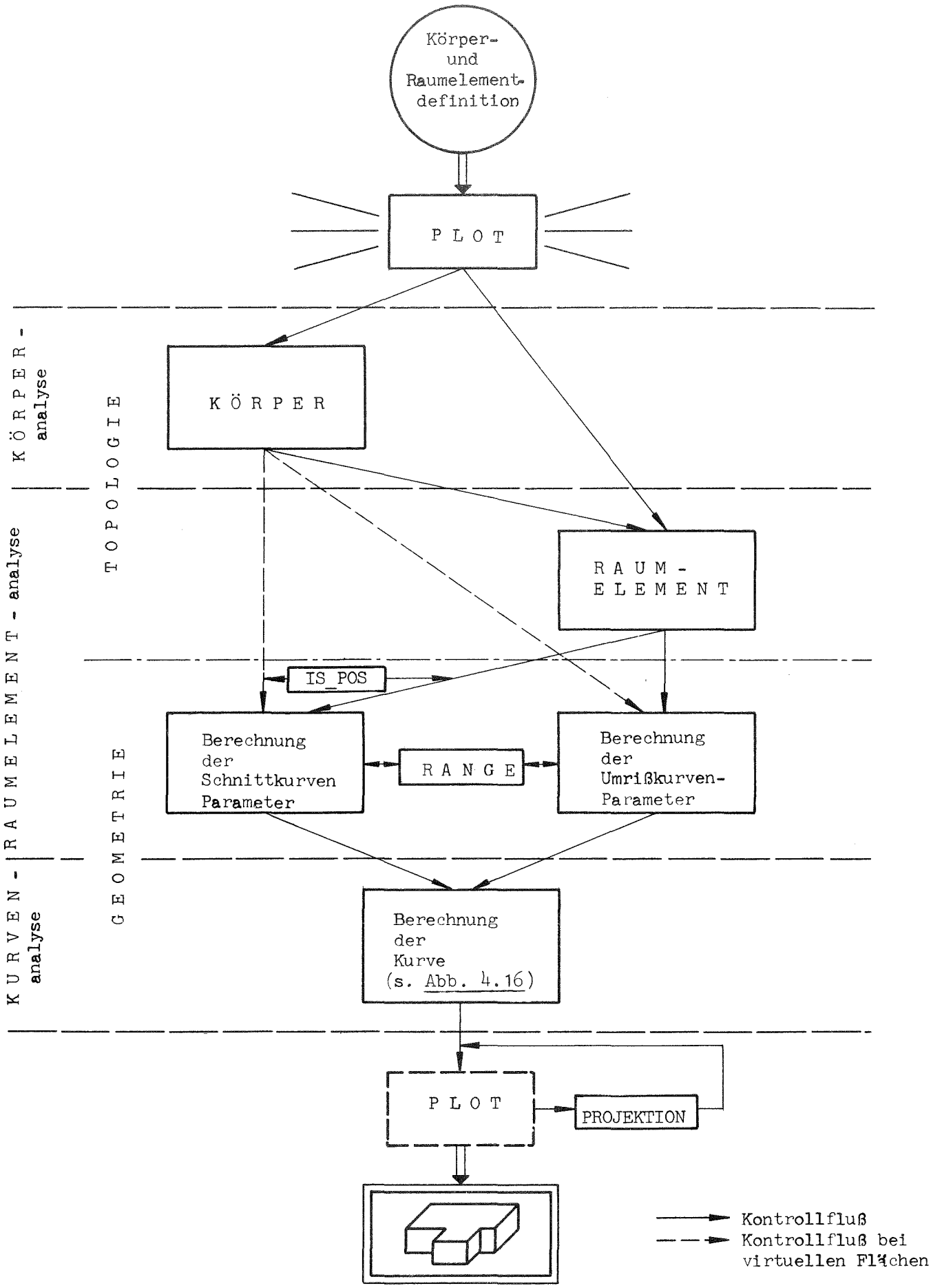


Abb. 4.15: Ablauf der Analyse räumlicher Objekte bei graphischer Ausgabe

Kurvenanalyse

Nach der vorgenannten Raumelementanalyse sind alle kurvenbestimmenden Konstanten und der Parameterbereich für eine explizite punktweise Berechnung einer Durchdringungs- oder Umrißkante anhand der Gleichungen aus Abb. 4.3 verfügbar. Die weitere Betrachtung kann sich auf die Untersuchung der Eigenschaften dieser Raumkurve hinsichtlich

- mathematischer oder
- materialler Existenz, sowie
- der Sichtbarkeit

konzentrieren (siehe Abb. 4.16).

Durch die inkrementale Variation des unabhängigen Parameters u wird analytisch eine Punktefolge auf der Raumkurve erzeugt. Dabei können die Eigenschaften der Kurve zu aufeinanderfolgenden Parameterwerten durch Überschreiten von Bereichsgrenzen zwischen

- reell und imaginär,
- zum Körpervolumen gehörig und
  nicht zum Körpervolumen gehörig oder
- sichtbar und nicht sichtbar

wechseln.

Die Entscheidung zur mathematischen Existenz eines Kurvenpunktes kann anhand des Radikanten der Wurzeln der Gleichungsformen (Abb. 4.3) für die Durchdringungen erfolgen. Nach dieser Prüfung, die für beide Kurvenäste (positives und negatives Vorzeichen der Wurzel) gleichzeitig gilt, wird über die Zugehörigkeit der (oder des) Raumpunkte(s) zum Körpervolumen (IS_MAT) entschieden.

Die Kurvenpunkte und ihre Abstände voneinander werden durch die Schrittweite der Parameteränderung bestimmt. Diese Inkremente beeinflussen in hohem Maße die Effektivität der Körperberechnung, da durch ihre Wahl die Anzahl der zu berechnenden und anlysierenden Raumpunkte abhängt.

Vom System werden als Schrittweite für gerade 8 mm und für gekrümmte Umriß- bzw. Durchdringungskurven 12 Grad angenommen. Für den Anwender besteht jedoch die Möglichkeit, diese Größen nach Darstellungs- und Effektivitätsgesichtspunkten anzupassen. Die Notwendigkeit zur Anpassung ergibt sich aus den Methoden des Körperaufbaues durch das Aneinanderfügen von Raumelementen, da nicht vorhersehbar ist, welche Kantenstücke in der

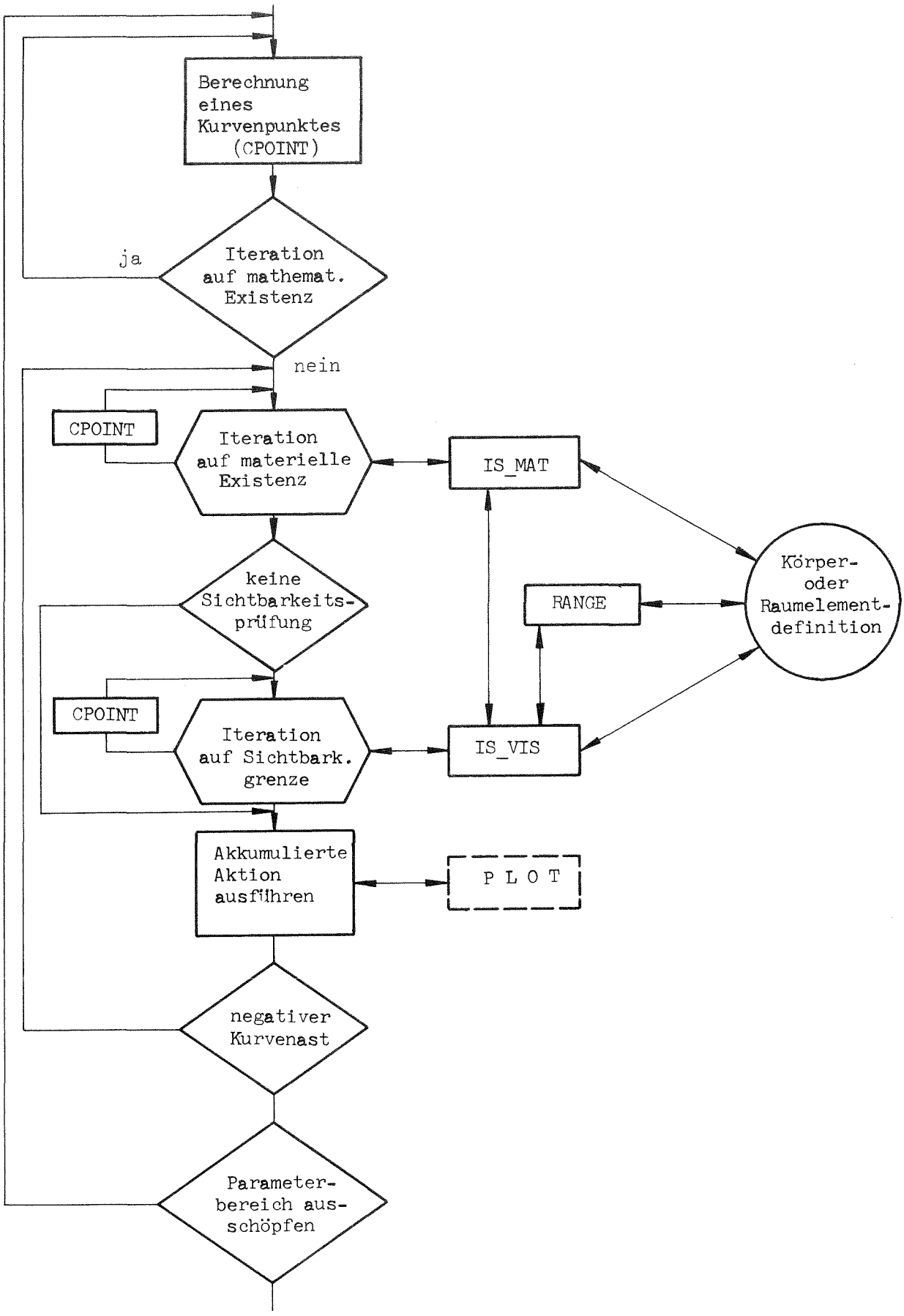


Abb. 4.16: Ablauf der Kurvenanalyse

Berührungsfläche entfallen. Die Prüfung der Charakteristiken der entstehenden Raumkurven erfolgt punktweise. Eigenschaftsänderungen, die sich nur auf Teilbereiche der Kurve zwischen zwei untersuchten Punkten auswirken, bleiben unerkannt. Das "Überspringen" solcher Kantenstücke muß durch die Spezifikation der erforderlichen Schrittweite vermieden werden.

Die vom Anwender gewählten Inkremente sollten aus Effektivitätsgründen so groß wie möglich im Hinblick auf eine wirklichkeitsgetreue Wiedergabe des Objektes jedoch so klein wie nötig sein.

Die Untersuchung von Kurvenpunkten auf Sichtbarkeit führt zu ähnlichen Problemen, wobei hier erschwerend die Abhängigkeit von der Betrachtungsrichtung hinzukommt. Es ist nicht voraussehbar, welche Teile einer Fläche sich auf welche Bereiche einer Raumkurve auswirken. Die Inkrementenwahl muß durch diese Abhängigkeit nicht nur objekt-, sondern u.U. auch projektionsbezogen erfolgen.

Der Algorithmus zur Entscheidung der Sichtbarkeit, der in einer Routine IS_VIS realisiert ist, interpretiert die in der Datenstruktur enthaltenen Körper- bzw. Raumelementdefinitionen unter Nutzung der Routinen IS_MAT für die Bestimmung der Körperzugehörigkeit und RANGE zur Errechnung der Durchtrittspunkte eines Sehstrahls. Auf die Einzelheiten dieser Untersuchung wird in Abschnitt 4.6 näher eingegangen.

Wird während des Entlangrechnens auf einer Raumkurve eine Änderung einer der Eigenschaften festgestellt, so wird mit der Iteration zur Berechnung des exakten Überganges begonnen. Die Ermittlung des zugehörigen Parameterwertes erfolgt über eine Schrittweitenhalbierung, die bei hinreichender Annäherung der untersuchten Punkte oder bei Unterschreitung eines minimalen Inkrementwertes abgebrochen wird.

Die Punktefolge der - je nach Schnittart ein oder zwei - Kurvenäste wird in Wertefeldern abgespeichert. Über die Ausgabe der Kurve wird nach Durchlaufen aller Stadien der Analyse anhand des akkumulierten Zustandes entschieden. Wird die zeichnerische Ausgabe eines Kurvenzuges veranlaßt, so erfolgt unmittelbar eine Abbildung der Raumkurve unter Nutzung der aktuellen Projektionsmatrix und der Darstellungsattribute für den behandelten Kantentyp. Die Speicherbereiche, die die Koordinatenwerte der Durchdringungs- oder Umrißkurve beinhalten, werden freigegeben und stehen für die Aufnahme eines neuen Kurvenstückes zur Verfügung.

Das für die Kantenbestimmung implementierte rechenzeiteffektive analytische Verfahren stellt damit auch geringe Anforderungen an den Speicherplatz, da der Prozeß der Kurvenanalyse sequentiell abläuft und jedes Kurvenstück unabhängig von anderen berechnet und in seinen Eigenschaften hinsichtlich Sichtbarkeit untersucht werden kann.

Virtuelle Flächen und Körperanalyse

Die bisherigen Erläuterungen schlossen die Behandlung von Körpern aus und konzentrierten sich auf die automatische Darstellung von Raumelementen. Nach Abb. 4.15 wird die Behandlung von Körpern ermöglicht durch eine gesonderte Betrachtung der virtuellen Flächen (vgl. Abschnitt 4.5). Dazu wird zunächst geprüft, ob die am darzustellenden Körper beteiligten Raumelemente Flächenpaarungen verwenden, die durch Umkehrung der Orientierung einer der schnittbildenden Flächen als 'virtuell' gekennzeichnet sind. Diese Flächenkombinationen werden in eine Tabelle eingetragen, die neben den Schnittpartnern diejenigen Raumelemente enthält, die sich entlang der virtuellen Fläche berühren.

Die Berechung der in dieser Tabelle vermerkten Schnittkurven erfolgt zunächst nach denselben Regeln wie vorher bei den Raumelementen, d.h. es werden die erforderlichen Kurvenparameter und die Bereichsgrenzen bestimmt, um damit die Kurvenanalyse zu beginnen. Die Überprüfung der materiellen Zugehörigkeit erfolgt jedoch für alle in der Tabelle gekennzeichneten Raumelemente. Da die betrachteten Raumelemente in der virtuellen Fläche zusammenstoßen, entfallen diejenigen Punkte bzw. Kantenstücke, die diese gemeinsam haben. Der Rest der Kurvenanalyse läuft wieder nach dem gleichen Formalismus wie bei Raumelementen ab.

Bei der Bestimmung von Umrißkurven für diejenigen Flächen, die als virtuell vermerkt sind, muß ebenfalls eine Überprüfung der Zugehörigkeit zu den beteiligten Raumelementen vorgenommen werden.

Im Beispiel der Abb. 4.11 ist als virtuelle Fläche der Zylinder L gegeben. Nach Abb. 4.14 sind von der Körperanalyse der Schnitt H-L und der Umriß L betroffen, die in der Matrix durch 'V' markiert sind.

Nach Abschluß der Körperuntersuchung läuft die Analyse der Reststruktur ab, bei der getrennt nach Raumelementen die Erzeugung der Abbildung vorgenommen wird. Die in der Körpertabelle enthaltenen und bereits berechneten Flächenpaarungen werden dabei allerdings ausgelassen.

Algorithmische Beschreibung von Körpern

Bisher wurde die Möglichkeit zur Behandlung von Körpern am Beispiel eines einfachen Objektes demonstriert, ohne auf die Fähigkeiten der Sprache zur algorithmischen Formulierung einzugehen. Abb. 4.17 zeigt ein Programm zur Darstellung eines Flansches. Unter Ausnutzung der Spracheigenschaften zur Feldvereinbarung und Schleifenbildung wird ein aus drei Raumelementen bestehender Körper spezifiziert.

```
SAMPLE:PROC OPTIONS(MAIN) REGENT(MOD=2,PLOT,POOL=140000,NODA);
 ENTER GIPSY;
    DCL FLANSCH COLL;
    DCL TEIL(4) SPACE(10);
    DCL RING(5) CYLINDER;
    DCL WAND(4) PLANE;
    DCL APUNKT(5) POINT;
    DCL CO2(5) DEC FLOAT(6) INIT(0.,10.,40.,70.,125.);
    DCL R(5) DEC FLOAT(6) INIT(70.,120.,60.,40.,20.);
     DO I=1 TO 5;
        SET APUNKT(I)=POINT(0.,CO2(I),0.);
     END;
     DO I=1 TO 4;
        SET RING(I) = CYLINDER(APUNKT(1),APUNKT(5),R(I));
        SET WAND(I) = PLANE(APUNKT(I),APUNKT(5));
     END;
     DO I=1 TO 3;
        SET TEIL(I) = SPACE(WAND(I)+RING(I)-RING(4)-WAND(I+1));
     END;
     SET RING(5)=SWITCH(CYLINDER(POINT(95.,0.,0.),POINT(95.,100.,0.),
                                 R(5)));
     DO I=0 TO 5;
        SET TEIL(4)=SPACE(TEIL(4)+ROTATE(RING(5),0.,I*60.,0.));
     END;
     SET FLANSCH=COLL(TEIL(1)+SPACE(TEIL(2)+TEIL(4))+TEIL(3));
     PRINT(FLANSCH);
     CHANGE  PROJECTION  PARALLEL  1.,  0.2,  0.2,
             PROJECTION  ORIGIN 120.,  150.;
     OPEN PLOT  DIN A3;
     PLOT(FLANSCH);
 END GIPSY;
END SAMPLE;
```

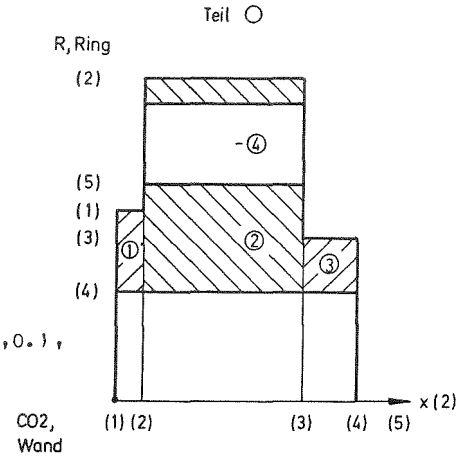


Abb. 4.17: Programmbeispiel: Flansch

Die geometrischen Abmessungen des Flansches, die in den Feldern CO2 und R gespeichert sind, lassen sich mit den Mitteln der Basissprache PL/1 leicht verändern, um aus einer gleichbleibenden Struktur der graphischen Befehle die Ausgabe eines Variantenteiles zu ermöglichen.

Ein derartiges Programm ist in einer Variantenbibliothek abspeicherbar und mit neuen geometrischen Abmessungen, die z.B. über Eingabeoperationen gewonnen werden, für eine andere Variante abrufbar.

Voraussetzung für die Ausnutzung der algorithmischen Fähigkeiten ist die Planung der Speicherung der das Objekt beschreibenden Flächen und geometrischen Angaben. Durch geschickte Wahl der Indizierung der Variablen für die Körperbeschreibung läßt sich eine kurze, übersichtliche und dazu fexible Problembeschreibung finden.

Neben der oben bereits diskutierten Möglichkeit zur Größenvariation von Bauteilen lassen sich durch geringe Programmänderungen Komplexteile darstellen. In Abb. 4.18 ist das Ausgangsobjekt Flansch gemäß Beschreibung in Abb. 4.17 einer Flanschverbindung gegenübergestellt. Die Darstellung des zweiten Flansches kann durch Spiegelung der Ebenen mit

```
SET WAND(I) = SCALE(WAND(I),1.,-1.,1.);
```

und Einbringung in ein zweites Objekt mit denselben Zylinderflächen erzeugt werden.

Zur Überprüfung der objektspezifizierenden Eingabe oder zu Protokollzwekken kann auch bei räumlichen Objekten die Druckausgabe der internen Datenstruktur in einer leicht lesbaren Form erfolgen. Diese Liste (Abb. 4.19) enthält neben den geometrischen Daten der körperbegrenzenden Flächen die Angabe über die Orientierung der Raumdefinitionen und die Aussagen über die strukturelle Zusammenfassung. Durch die Prüfung dieser Angaben ist es möglich, Testzeit und damit Kosten zu sparen, da einige Fehlversuche für die aufwendigere Körperanalyse der graphischen Darstellung (Abb. 4.18) vermieden werden können.

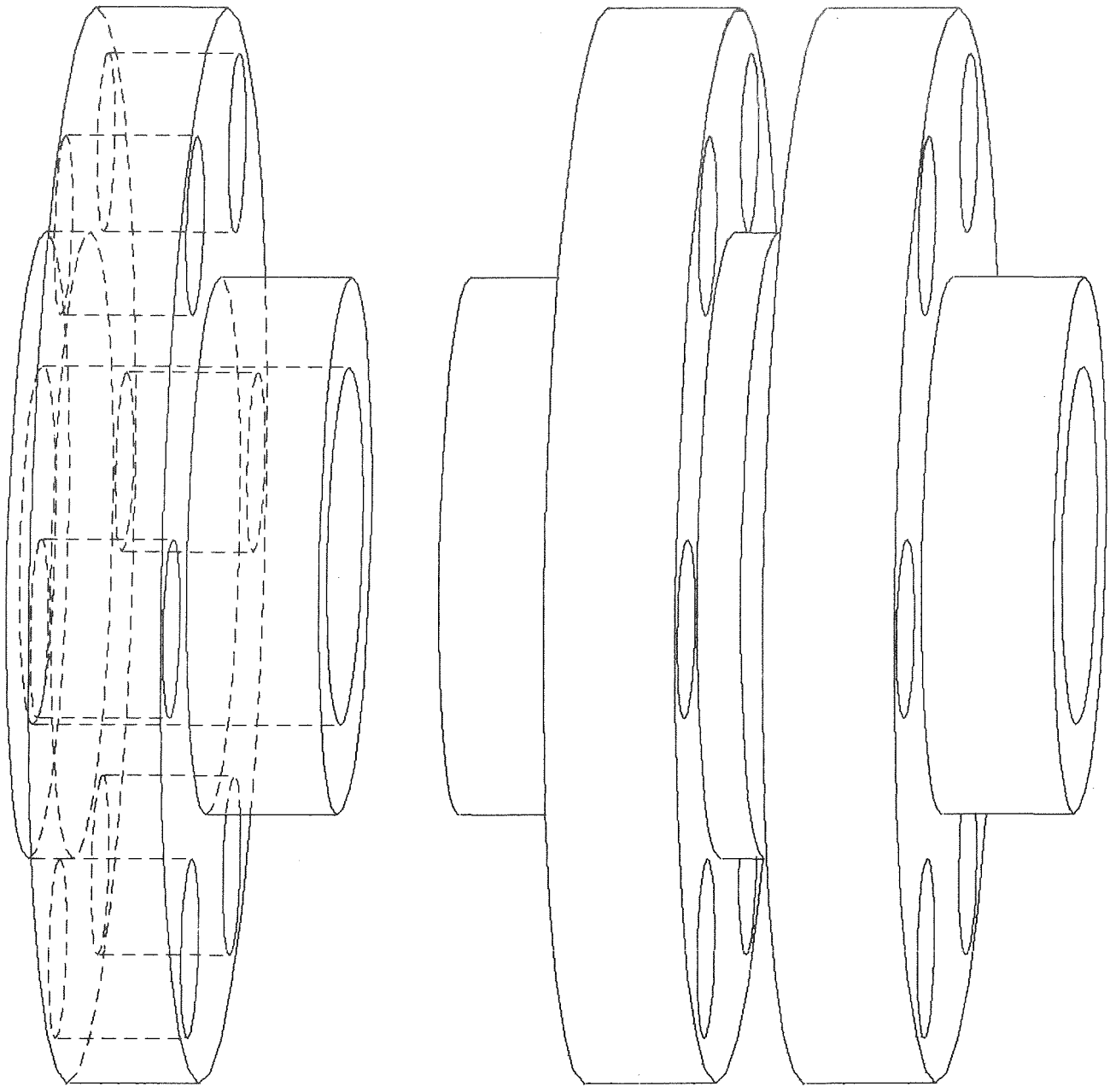


Abb. 4.18: Graphische Ausgabe des Flansches

```
+----------------------------------------------------------------------+
| COLLECTION:FLANSCH                                                   |
+----------------------------------------------------------------------+
| SPACE:1.SUBOBJECT OF FLANSCH                                         |
+----------------------------------------------------------------------+
| PLANE:1.SURFACE OF 1.SUBOBJECT OF FLANSCH                            |
+----------------------------------------------------------------------+
| XO(*)   | 0.000000E+00      0.000000E+00      0.000000E+00           |
| NORMAL  | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE                                                 |
+----------------------------------------------------------------------+
| CYLINDER:2.SURFACE OF 1.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  | 0.000000E+00      0.000000E+00      0.000000E+00           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE          RADIUS: 6.999999E-02                   |
+----------------------------------------------------------------------+
| CYLINDER:3.SURFACE OF 1.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  | 0.000000E+00      0.000000E+00      0.000000E+00           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE          RADIUS: 4.000000E-02                   |
+----------------------------------------------------------------------+
| USED IN OPPOSITE DIRECTION                                           |
+----------------------------------------------------------------------+
| PLANE:4.SURFACE OF 1.SUBOBJECT OF FLANSCH                            |
+----------------------------------------------------------------------+
| XO(*)   | 0.000000E+00      9.999996E-03      0.000000E+00           |
| NORMAL  | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE                                                 |
+----------------------------------------------------------------------+
| USED IN OPPOSITE DIRECTION                                           |
+----------------------------------------------------------------------+
| SPACE:2.SUBOBJECT OF FLANSCH                                         |
+----------------------------------------------------------------------+
| PLANE:1.SURFACE OF 2.SUBOBJECT OF FLANSCH                            |
+----------------------------------------------------------------------+
| XO(*)   | 0.000000E+00      9.999998E-03      0.000000E+00           |
| NORMAL  | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE                                                 |
+----------------------------------------------------------------------+
| CYLINDER:2.SURFACE OF 2.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  | 0.000000E+00      0.000000E+00      0.000000E+00           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE          RADIUS: 1.199999E-01                   |
+----------------------------------------------------------------------+
| CYLINDER:3.SURFACE OF 2.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  | 0.000000E+00      0.000000E+00      0.000000E+00           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE          RADIUS: 4.000000E-02                   |
+----------------------------------------------------------------------+
| USED IN OPPOSITE DIRECTION                                           |
+----------------------------------------------------------------------+
| PLANE:4.SURFACE OF 2.SUBOBJECT OF FLANSCH                            |
+----------------------------------------------------------------------+
| XO(*)   | 0.000000E+00      4.000000E-02      0.000000E+00           |
| NORMAL  | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE                                                 |
+----------------------------------------------------------------------+
| USED IN OPPOSITE DIRECTION                                           |
+----------------------------------------------------------------------+
| CYLINDER:5.SURFACE OF 2.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  | 9.499997E-02      0.000000E+00      0.000000E+00           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED OUTSIDE         RADIUS: 2.000000E-02                   |

+----------------------------------------------------------------------+
| CYLINDER:6.SURFACE OF 2.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  | 4.750003E-02      0.000000E+00     -8.227235E-02           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED OUTSIDE         RADIUS: 2.000000E-02                   |
+----------------------------------------------------------------------+
| CYLINDER:7.SURFACE OF 2.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  |-4.750002E-02      0.000000E+00     -8.227235E-02           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED OUTSIDE         RADIUS: 2.000000E-02                   |
+----------------------------------------------------------------------+
| CYLINDER:8.SURFACE OF 2.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  |-9.499997E-02      0.000000E+00      0.000000E+00           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED OUTSIDE         RADIUS: 2.000000E-02                   |
+----------------------------------------------------------------------+
| CYLINDER:9.SURFACE OF 2.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  |-4.749989E-02      0.000000E+00      8.227241E-02           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED OUTSIDE         RADIUS: 2.000000E-02                   |
+----------------------------------------------------------------------+
| CYLINDER:10.SURFACE OF 2.SUBOBJECT OF FLANSCH                        |
+----------------------------------------------------------------------+
| CENTER  | 4.750005E-02      0.000000E+00      8.227229E-02           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED OUTSIDE         RADIUS: 2.000000E-02                   |
+----------------------------------------------------------------------+
| SPACE:3.SUBOBJECT OF FLANSCH                                         |
+----------------------------------------------------------------------+
| PLANE:1.SURFACE OF 3.SUBOBJECT OF FLANSCH                            |
+----------------------------------------------------------------------+
| XO(*)   | 0.000000E+00      4.000000E-02      0.000000E+00           |
| NORMAL  | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE                                                 |
+----------------------------------------------------------------------+
| CYLINDER:2.SURFACE OF 3.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  | 0.000000E+00      0.000000E+00      0.000000E+00           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE          RADIUS: 5.999999E-02                   |
+----------------------------------------------------------------------+
| CYLINDER:3.SURFACE OF 3.SUBOBJECT OF FLANSCH                         |
+----------------------------------------------------------------------+
| CENTER  | 0.000000E+00      0.000000E+00      0.000000E+00           |
| DIRECT. | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE          RADIUS: 4.000000E-02                   |
+----------------------------------------------------------------------+
| USED IN OPPOSITE DIRECTION                                           |
+----------------------------------------------------------------------+
| PLANE:4.SURFACE OF 3.SUBOBJECT OF FLANSCH                            |
+----------------------------------------------------------------------+
| XO(*)   | 0.000000E+00      6.999999E-02      0.000000E+00           |
| NORMAL  | 0.000000E+00      1.000000E+00      0.000000E+00           |
+----------------------------------------------------------------------+
| SPACE DEFINED INSIDE                                                 |
+----------------------------------------------------------------------+
| USED IN OPPOSITE DIRECTION                                           |
+----------------------------------------------------------------------+
```

Abb. 4.19: Druckausgabe des Flansches

4.6 Sichtbarkeitsalgorithmus

Die Computer-Graphik ist bemüht, von räumlichen Objekten realistische, der Wahrnehmung (oder Zeichnungsnormen) entsprechende Darstellungen zu ermöglichen. Bei der Wiedergabe eines Objektes müssen nach einer Berechnung der räumlichen Koordinaten der Kanten die Probleme der Projektion und der Sichtbarkeit gelöst werden. Letztere stellen aufgrund der Anforderungen an die Rechenzeit und die Programmlogik - auch heute noch - ein Hauptarbeitsgebiet der Computer-Graphik dar.

In Abschnitt 2 wurden einige Algorithmen zur Behandlung des Sichtbarkeitsproblems untersucht. Das Bestreben jeder Implementierung ist es, den rechnerischen Aufwand für die Sichtbarkeitsentscheidung zu reduzieren. Diese Effektivitätssteigerungen wurden in fast allen Fällen erkauft durch eine Einschränkung des behandelbaren Objektspektrums (Polyeder) oder die Festlegung auf eine starre Bildebene.

Bei Algorithmen zur Behandlung von Polyedern herrscht die Tendenz vor, die Sichtbarkeitsentscheidung im Bildraum vorzunehmen. Adaptionen dieser Verfahren der 'quantitative invisibility' und der 'edge classification' auf Körper mit Flächen höherer Ordnung /20,164/ zeitigen nicht die gleichen Vorteile gegenüber den Punktetests wie bei Polyedern, da die Folgerungen aus scheinbaren Schnitten von Körperkanten in der Bildebene nicht nur einmal für die gesamte Kante gezogen werden. Die Untersuchung von gekrümmten Kantenverläufen muß sich auf alle Segmente erstrecken. Für diese Kurvenform ist dabei neben der längeren Rechenzeit für die Schnittfindung zwischen Kanten auch ein erheblich größerer Speicheraufwand erforderlich, da eine Sichtbarkeitsentscheidung erst vorgenommen werden kann, wenn die Koordinatenpaare aller Bildpunkte berechnet und in gleichzeitig verfügbaren Feldern abgespeichert sind.

Im vorliegenden System wurde ein Algorithmus implementiert, der alle Sichtbarkeitsentscheidungen im Objektraum trifft. Für die Abbildung der Objekte ergeben sich keine Einschränkungen hinsichtlich Lineartransformationen, Lage der Bildebene und Wahl der Projektionsart oder -richtung. Trotz dieser Flexibilität ist das Verfahren kein reiner Punktetest wie bei /88/, /167/ und /142(?)/, sondern eine Kombination aus:

- Flächen-
- Kanten- und
- Punktetest.

Die bereits früh einsetzenden zusätzlichen Flächen- und Kantentests wirken sich effektivitätssteigernd aus.

Ein Flächentest ist nur auf Ebenen anwendbar. Ebenen - und die mit ihnen gebildeten Kanten - sind potentiell sichtbar, wenn ihre Normalen $\underline{n}_E$ mit dem Betrachtungsvektor $\underline{v}$ einen Winkel $< 90^\circ$ bildet, d.h. das Skalarprodukt $\underline{n}_E \, \underline{v}$ ist positiv. Das Ergebnis dieser Berechnung wird zur Kennzeichnung jeder Fläche abgespeichert und ist für die weitere Untersuchung verfügbar.

Bei Kantentests erfolgt die Sichtbarkeitsaussage ohne weitere Untersuchung für den gesamten Kurvenverlauf:

- Die Kantenberechnung erfolgt durch Schnittbildung von zwei Flächen. Sind beide Schnittpartner Ebenen und als unsichtbar gekennzeichnet, so ist auch die mit ihnen gebildete Kante unsichtbar ('edge classification').
- Umrißkanten von Flächen, deren Raumdefinition durch Inversion oder Umkehrung nach außen zeigt (Hohlflächen), sind unsichtbar.

Der Punktetest muß zur Untersuchung derjenigen Kurvenzüge herangezogen werden, über deren Sichtbarkeit nicht aufgrund einer der obigen Aussagen entschieden werden konnte.

- Für Punkte auf Schnittkurven zwischen Flächen 2. Ordnung kann in Analogie zur 'edge classification' eine Aussage über die Unsichtbarkeit dadurch gemacht werden, daß die Normalen der beteiligten Flächen in diesem Punkt hinsichtlich ihres Winkels bezüglich des Betrachtungsvektors untersucht werden. Es ist so möglich, eine - allerdings punktweise - Festlegung der Unsichtbarkeit ohne eine Überprüfung auf Fremdverdeckung vorzunehmen.

  Die im Anhang F4. zusammengestellten Gleichungen bilden die Grundlage dieser Entscheidung.

- Für den Rest der Punkte verbleibt ein Punktetest, der dann zur Feststellung der Unsichtbarkeit führt, wenn der Betrachtungsvektor den Definitionsbereich des Körpers durchdringt. Die Durchstoßpunkte des Betrachtungsstrahls durch die Körperoberfläche können mit Hilfe der Routinen RANGE und IS_MAT, die bereits bei der 'materiellen' Untersuchung der Körper verwendet wurden (vgl. Abb. 4.15 und Abb. 4.16), ermittelt werden. Es sei daran erinnert, daß diese Berechnungen ebenfalls auf analytisch ermittelten Beziehungen basieren.

  Prinzipiell müssen alle körperbegrenzenden Flächen zur Untersuchung herangezogen werden, aufgrund der speziellen Implementierung gelten folgende verfahrensbeschleunigende Zusatzbedingungen:

  . schnittbildende Ebenen verdecken nicht ihre Kante,
  . umrißbildende Flächen verdecken nicht ihre Kontur und
  . unsichtbare Ebenen verdecken nicht.

Aus diesen Bedingungen folgt auch, daß keine numerischen Probleme aus mangelnder Rechengenauigkeit entstehen, da diese Abfragen sich nicht auf arithmetisch gewonnene Werte beziehen, sondern über die Identitäten der Flächen entschieden werden.

Ein weiterer Vorteil des hier implementierten Sichtbarkeitsverfahrens liegt in der Möglichkeit seiner Anwendung auf vom verdeckenden Körper unabhängigen Strukturen. Es können so zusätzliche Linien in eine Abbildung unter Prüfung auf Sichtbarkeit eingebracht werden, um die Anschaulichkeit z.B. bei Architekturzeichnungen zu verbessern.

Die Fähigkeiten des Sichtbarkeitsalgorithmus wurden bereits durch einige Beispiele im bisherigen Text, in denen unsichtbare Kanten behandelt wurden, ersichtlich. Eine exemplarische Demonstration soll in Abb. 4.20 erfolgen, in der die Ausgabe des Raumelementes aus Abb. 4.8 und Abb. 4.9 bei fortwährender Rotation unter Vernachlässigung der unsichtbaren Kanten dargestellt ist. Die maßgeblichen Statements des zugehörigen Programmes lauten:

```
SET WUERFEL = SHIFT (WUERFEL,-50.,-50.,-50.);
CHANGE LINETYPE INVISIBLE OMITTED;
SET KUGEL = BALL (POINT(0.,0.,0.),65.);
DO  AL = 0. TO  80.  BY  9.;
    OPEN PLOT;
    PLOT (ROTATE(SPACE(WUERFEL-KUGEL),AL,0.,0.));
END;
```

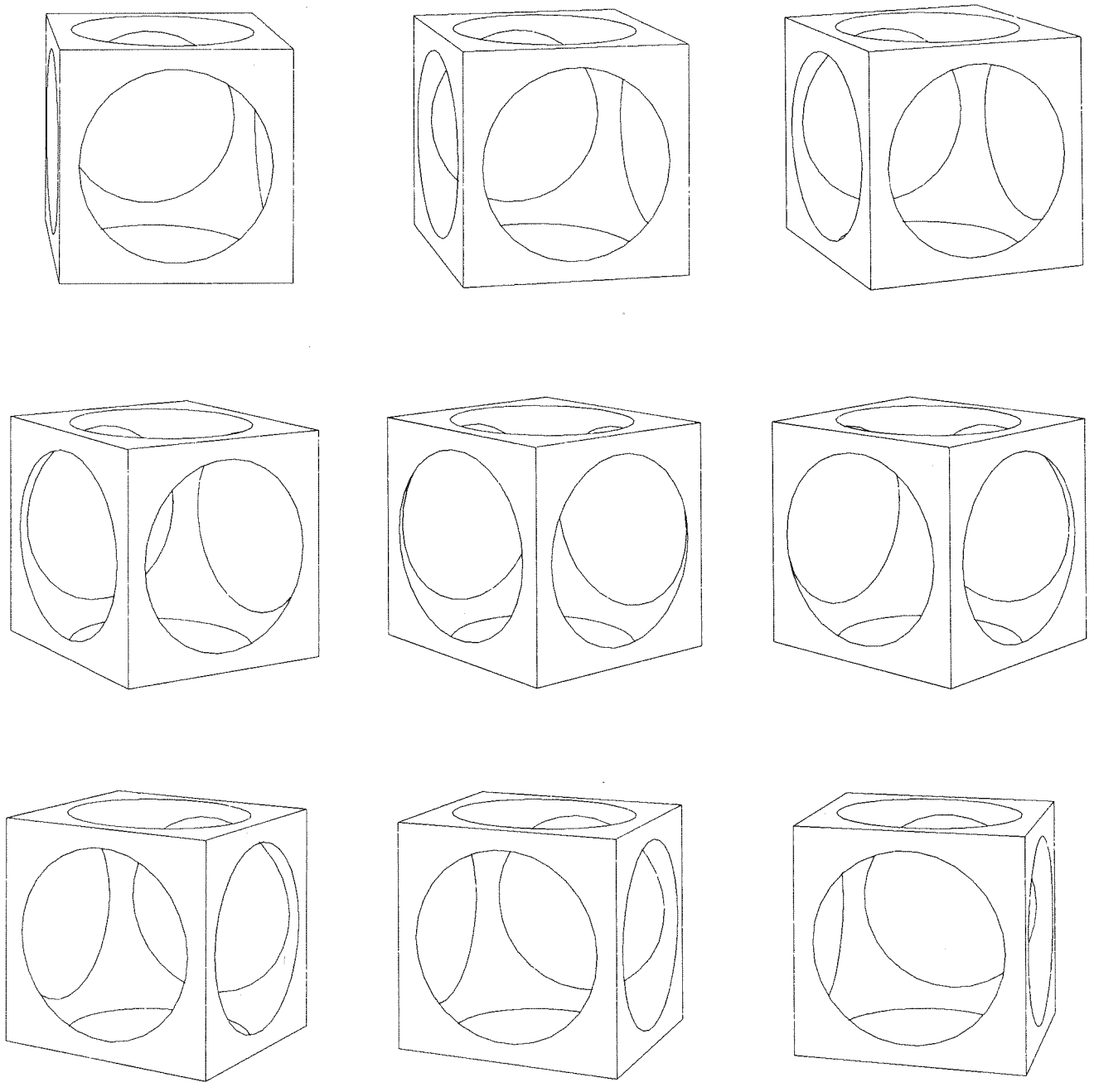


Abb. 4.20: Abbildungen eines Objektes unter Vernachlässigung der unsichtbaren Kanten

4.7 Darstellungshilfen durch Ausschnitte, Explosionsanordnung und Schnitte.

Eine technische Zeichnung enthält eine Vielzahl von Informationen zur Geometrie, zur Bearbeitung, zur Montage, usw., für deren zweifelsfreie Interpretation unterschiedliche Methoden der Verdeutlichung angewendet werden.

Die folgenden Verfahren besitzen bei der graphischen Darstellung besondere Bedeutung:

- Ausschnitte,
- Explosionsanordnungen und
- Schnitte.

Ausschnitte

Bei der Beschreibung von Konstruktionsobjekten sind Abbildungen zu erzeugen, die in ihrem Detaillierungsgrad variieren. Während in einer Zusammenstellungszeichnung ein Gesamtüberblick über eine Baugruppe gegeben wird, sind in den zugehörigen Einzteilzeichnungen alle zu deren Fertigung erforderlichen Angaben einzutragen. Diese Darstellungen erfolgen mit unterschiedlicher Auflösung durch die Auswahl von Teilbereichen.

Mit den Mitteln der GIPSY-Sprache lassen sich Ausschnitte auf zwei Wegen spezifizieren:

- Durch eine Auswahl bestimmter Namen oder Indizes kann die Ausgabe auf allein interessierende Objekte beschränkt werden.
- Der interessierende Objektsbereich läßt sich durch Einschluß mittels zusätzlicher Flächen ausblenden. Im einfachsten Fall reicht die Verwendung einer Kugel als einschließendes Volumen aus. Diese zusätzlichen Flächen sind nach den allgemeinen Regeln manipulierbar, so daß der interessierende Ausschnitt schrittweise über das Objekt geführt werden kann.

Explosionsanordnung

Zur Erstellung von Montageunterlagen wird häufig eine Explosionsdarstellung von Bauteilen gefordert. Die in GIPSY realisierte Technik des Aufbaues von Körpern ermöglicht hierzu die unabhängige Anordnung von Raumelementen bzw. Körpern im Bildraum. Die erforderliche Verschiebung der Abbildung von Teilobjekten gegeneinander kann durch folgende Statementfolge bewirkt werden:

```
DO  I = 1  TO N;
    CHANGE PROJECTION ORIGIN  I*DX,I*DY;
    PLOT (TEIL(I)),
END;
```

Durch diese Veränderung des Koordinatenursprungs im Bildraum würden die verschiedenen Teile eines Körpers entlang einer Geraden versetzt dargestellt.

Schnitte

Als wichtigste Darstellungshilfe in technischen Abbildungen dient die Anbringung eines Schnittes durch das darzustellende Werkstück mit Schraffieren der dabei entstehenden Körperoberfläche. Ist das räumliche Objekt durch eine hierarchische Datenstruktur aus Punkten, Kanten und Flächen aufgebaut, so wird durch die Schnittoperation eine Änderung dieser Struktur erzwungen, um die neuentstehenden Objekte der Trennfläche aufnehmen zu können /71/. Da im vorliegenden System die Werkstückkanten nicht als starre Datenstruktur, sondern als algorithmisch interpretierbare Erzeugungsvorschriften vorliegen, können auch Schraffuren in Schnittflächen auf diese Weise bestimmt werden.

Die Erzeugung eines Schnittes wird durch Angabe eines PLOT-Statements (vgl. Abschnitt 3.3.4.3 und Anhang A) bewirkt:

```
PLOT  SHADE [OF] < grevar> [WITH < grevar>]
      [IN] SURFACES ( < grevar >[, < grevar>]*);
```

Die Berechnung des Schnittes erfolgt dabei durch das Überstreichen dieser Fläche mit einer schnittbildenden Ebene.

Als Schnittflächen können alle verfügbaren Flächentypen angegeben werden.

Durch diese Angaben wird die Schraffur einer Fläche eines Raumelementes oder Körpers veranlaßt. Die ausgewählte Fläche kann eine der begrenzenden Oberflächen oder eine zusätzliche das Volumen schneidende Fläche sein.

Im Beispiel der Abb. 4.21 wurde die Deckebene D des Zylinders von TEIL (1) (s.Abb. 4.11) als Schnittfläche gekennzeichnet und gleichzeitig in einem Loop verschoben.

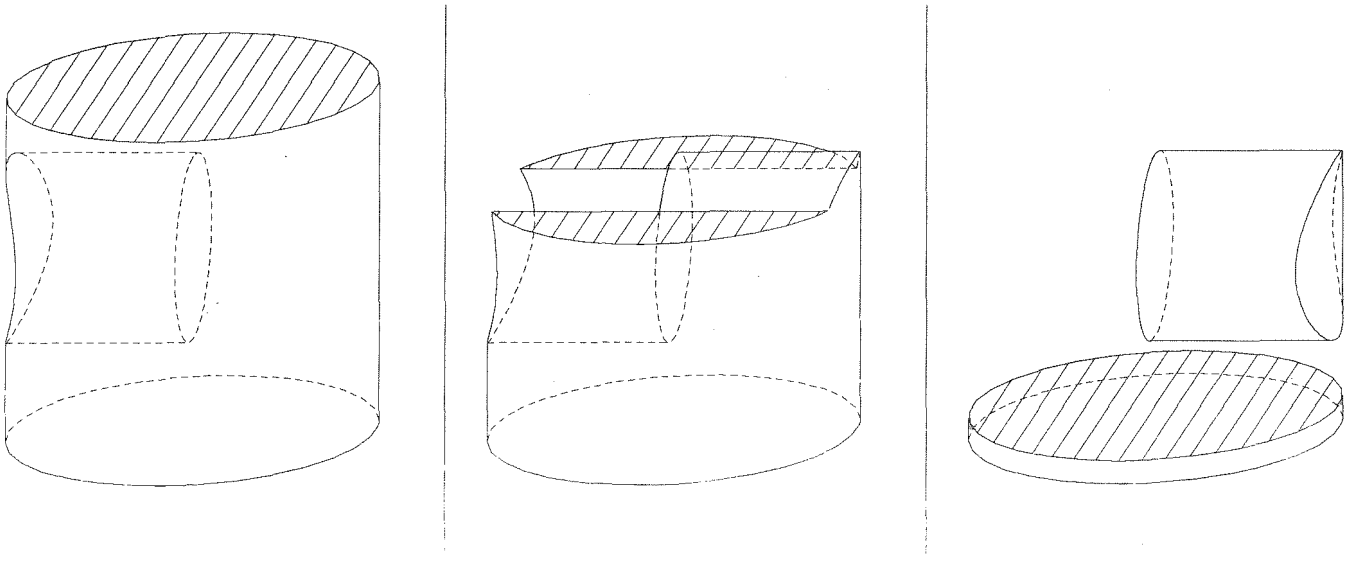


Abb. 4.21: Körper mit Kennzeichnung einer Schnittfläche

5. Betrachtung der graphischen Fähigkeiten des Systems GIPSY unter Aspekten des Konstruktionsprozesses

Die Konstruktionstätigkeit ist in den letzten zehn Jahren einer wissenschaftlichen Analyse unterzogen worden. Diese Untersuchungen führten zu einer systematischen Gliederung der Methoden des Konstruktionsprozesses sowie zu quantitativen Aussagen über die Verteilung der dabei ausgeführten Tätigkeiten. Der rechnerunterstützte Entwurf versucht, diese Erkenntnisse dem Einsatz moderner Datenverarbeitungstechniken zugänglich und damit dem Konstrukteur als Werkzeug nutzbar zu machen.

Der allgemeine wirtschaftliche Druck zwingt zur Rationalisierung in allen Produktionsbereichen. Die Untersuchungen zeigen, daß gerade in konstruktiven Bereichen ein erheblicher Nachholbedarf besteht. So wurde festgestellt, daß die Produktivitätssteigerung bei konstruktiven Tätigkeiten seit der Jahrhundertwende nur ca. 20 % betrug, während in der Fertigung der etwa 50-fache Anstieg verzeichnet werden konnte /141/. Obwohl sich an den Hilfsmitteln und Methoden des Konstrukteurs wenig geändert hat, muß heute ein breiteres Produktspektrum bei kürzerer Innovationszeit unter verschärften Anforderungen bearbeitet werden.

Es ist daher nicht verwunderlich, daß der Konstruktion die Verantwortung als wichtigster zeitverursachender Faktor zukommt. Bei Unternehmen mit Einzelfertigung wurden die Durchlaufzeiten bei 50 bis 70 % der Produkte von Tätigkeiten in Konstruktion und Arbeitsvorbereitung bestimmt /23/. Der Gesamtprozeß gliedert sich dabei in die Phasen:

- Funktionsfindung,
- Prinziperarbeitung,
- Gestaltung und
- Detaillierung.

In der Funktionsfindungsphase wird die Funktion eines Produktes aus den Anforderungen der Kunden (des Marktes) und den Fähigkeiten und Erfahrungen des Herstellers ermittelt und in einer Beschreibung der Eigenschaften (Pflichtenheft) festgehalten. Diese Tätigkeit, die auch noch stark von Wirtschaftlichkeitsüberlegungen beeinflußt wird, hat vorwiegend heuristischen Charakter und ist daher dem Rechnereinsatz kaum zugänglich. Ähnliches gilt für die Prinziperarbeitung, in der für die geforderte Gesamtfunktion eine Teilmenge der möglichen Arbeitsprizipien so zu strukturieren ist, daß sie der Aufgabenstellung gerecht wird /91,92/. Diese Auswahl

umfaßt häufig nicht mehr alle technisch möglichen Lösungen, sondern ist durch den Ausschluß bestimmter Funktionselemente oder Strukturen eingeengt.

In der nachgeschalteten Gestaltungsphase werden den Elementen Eigenschaften zugeordnet, die sich aus der gewünschten Funktion ergeben. Der Übergang zur Detaillierung ist bei hinreichender Gliederung nach funktionalen oder fertigungstechnischen Gesichtspunkten fließend und mündet in der Dimensionierung oder Darstellung von Bauteilen.

In allen Phasen ist die Beschaffung, Speicherung und Wiedergabe von Informationen eine Hauptaufgabe. Universelles Kommunikationsmedium ist dabei die Zeichnung. In den ersten Phasen handelt es sich jedoch vorwiegend um Freihandskizzen, die der Selbstdokumentation von Ideen zu Lösungsansätzen oder der Stützung der eigenen Anschauung dienen. Dieses Skizzieren geht in den Phasen des Gestaltens und Detaillierens in eine Zeichentechnik über, die sich innerhalb fester, durch Normen festgelegter Grenzen bewegt.

Diese Arbeiten, die nach verschiedenen aufgenommenen Tätigkeitsprofilen zwischen 25 und 37 % des zeitlichen Aufwandes im Konstruktionsbereich ausmachen /17,23,159/, werden durch einen hohen Anteil an manuell schematischen Vorgängen bestimmt und sind deshalb für eine Rechneranwendung prädestiniert.

Der in diesen Bereichen erzielbare Rationalisierungseffekt wird deshalb stark von der Erleichterung oder der vollständigen Automatisierung der Zeichnungserstellung abhängen. Es ist daher zu untersuchen, wie weit die graphischen Fähigkeiten des Systems GIPSY die zeichnerischen Tätigkeiten ergänzen oder substituieren. Die Merkmale einer vom Rechner erzeugten Zeichnung lassen sich in ähnlicher Weise wie bei konventionellen Zeichentechniken /156/ ordnen nach:

- Modellbildung,
- Darstellungshilfe,
- Informationsträger und
- Zeichentechnik.

Der Konstruktion kommt neben der maßgeblichen Beeinflussung der Durchlaufzeit eines Produktes auch die Hauptverantwortung für seinen Preis zu. Zwar ist ihre Auswirkung auf die verursachten Kosten mit 20 % gering, betrachtet man aber die festgelegten Kosten, so ist der Anteil des konstruktiven Bereiches mit fast 75 % beherrschend /23/. Dieser ergibt sich aus der für den weiteren Produktionsprozeß bestimmenden Auswahl des Prinzips, der Materialien, der Fertigungstechnik, usw.. Ziel muß daher die optimale Festlegung aller Produkteigenschaften als Ergebnis einer Wertanalyse sein. Diese Auswahl und Bewertung kann bei komplexen Objekten nur mit dem Rechner durchgeführt werden.

Für die graphischen Fähigkeiten ergibt sich daraus die Notwendigkeit für ihre Einbettung in ein umfassendes System zur rechnerunterstützten Lösung aller Teilschritte und -aufgaben des Entwurfsprozesses.

Eine Produktivitätssteigerung durch die Nutzung eines graphischen Systems muß nach dem oben Ausgeführten auf zwei Wegen angestrebt werden:

- Durch die Entlastung der Mitarbeiter von schematischer Tätigkeit wird ein ständiger zeitlicher Engpaß beseitigt und damit auch Arbeitskraft für kreatives Wirken freigesetzt.

- Durch die frühzeitige Kopplung von Programmen zur optimalen Auslegung von Produkten hinsichtlich verschiedenster Eigenschaften mit den Moduln zu ihrer Darstellung und der Erzeugung von Fertigungsunterlagen über eine gemeinsame Datenbasis ohne manuellen Eingriff.

Im folgenden sollen die Eigenschaften des Systems GIPSY unter diesen Gesichtspunkten näher betrachtet werden.

5.1 Merkmale der Zeichentätigkeit

5.1.1 Modellbildung

Die Formulierung von technischen Zusammenhängen erfolgt häufig in abstrahierenden Modellen. Auch eine Zeichnung ist das Modell eines Objektes, das dabei in seinen Eigenschaften bezüglich

- der Funktion,
- der Struktur,
- der Form,
- des Materials, usw.

beschrieben wird.

Beim konventionellen Zeichnen entsteht graphische Information immer direkt auf einem Zeichenmittel (Papier, Tafel, usw.). Die mittelbare Erstellung von Zeichnungen mit einer an einen Rechner angeschlossenen Zeichenmaschine erfolgt nach einer Speicherung in elektrischen Zuständen. Zur Programmierung dieser Zustände stehen die Eingangskanäle des Rechners zur Verfügung. Sieht man von den später noch zu diskutierenden interaktiven Techniken ab, so kommt nur eine Formulierung der graphischen Aufgaben in einer problemorientierten Sprache in Frage.

Die sprachliche Beschreibung der geometrischen Eigenschaften von Objekten bedingt einen hohen Eingabeaufwand. In der Literatur findet sich eine Anzahl von Beschreibungssystemen /15,71,80,86,141/, die - vorwiegend für rotationssymmetrische Werkstücke - die Eingabe durch Code-Wörter für häufig auftretende Formelemente reduzieren helfen. Zielsetzung aller dieser Systeme ist mehr die Erstellung von vollständigen Werkstattzeichnungen für diese spezielle Objektauswahl als die Verbreiterung der Basis zur Behandlung allgemeiner graphischer Probleme. D.h. sie sind als abgeschlossene Systeme zu betrachten, das es i.a. nicht ermöglicht, perspektivische Darstellungen zu erhalten oder algorithmische Beschreibungen der Objekte vorzunehmen.

In den Fällen, wo Körper beschrieben und räumlich abgebildet werden können, beschränkt sich der Objektumfang auf Polyeder /71,80/.

Im vorliegenden System wird ganz allgemein nach der Bereitstellung von Fähigkeiten zur Behandlung vertrauter zeichnerischer Tätigkeiten mit den Mitteln der graphischen Datenverarbeitung gestrebt. Unter dem Gesichtspunkt einer effektiven Objektbeschreibung sind hierzu folgende Eigenschaften zu nennen:

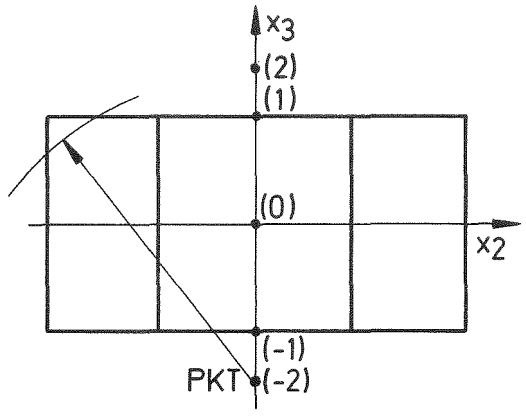
- eine problemorientierte Sprache mit allen algorithmischen Fähigkeiten,
- ein Objektumfang, der den graphischen Problemen angemessen ist, und
- eine große Modularität.

Algorithmische Fähigkeiten

Die Behandlung von Zeichnungsaufgaben mit dem Computer verlangt eine genauere Planung der Zeichentätigkeit. Ein Konstruktur entwickelt die Darstellung eines Objektes von einigen Fixpunkten oder -linien ausgehend in zweidimensionalen Ansichten, die, solange er sich an bestehende Konventionen (Normen) hält, ein zweifelsfreies Kommunikationsmittel ist, da er während des Zeichenvorganges - fast unbewußt - Informationen über den Verlauf oder die Sichtbarkeit von Körperkanten bestimmt und einträgt. Das Synthetische dieses Prozesses ist dem Zeichnenden auf dieser elementaren Ebene im Gegensatz zum Aufbau von Komplexen aus bereits vorhandenen Grundkörpern meist nicht bewußt.

Dem Rechner müssen vor Beginn der Analyse eines darzustellenden Körpers alle Angaben über die exakte Lage, die Gestalt und die gegenseitige Zuordnungen von Teilobjekten vorliegen, um aus der generierten Datenstruktur unter Anwendung effektiver Algorithmen eine Modellierung des Objektes vornehmen zu können.

Die algorithmischen Eigenschaften der GIPSY-Sprache erlauben die Spezifikation dieser Angaben in Abhängigkeit von arithmetischen oder logischen Ausdrücken in flexibelster Weise. Die Form und die Abmessungen von Objekten können dadurch in den sie beschreibenden Routinen variabel gehalten werden. Die Möglichkeiten der algorithmischen Beschreibung von Objekten soll anhand des Kopfes einer Sechskantschraube (s. Abb. 5.1) dargelegt werden, deren Programmierung für beliebige Schlüsselweiten SW wie folgt vorgenommen werden kann:




```
DCL  SKS  SPACE(10),
     PKT(-2:2) POINT;
     E=1.155 * SW;
     DO J=1, -1;
       DO I=1 TO 2;
          SET PKT(I*J)=POINT(0.,0.,F(I,J,SW));
          IF I=1 THEN
             SET SKS=SPACE(SKS+PLANE(PKT(I*J),PKT(0)));
          ELSE
             SET SKS=SPACE(SKS+BALL(PKT(I*J),E*0.73));   /* ABRUNDUNG */
       END;
     END;
     DO  I=1 TO 6;
         SET SKS=SPACE(SKS+PLANE(ROTATE                  /* SECHSKANT */
                 (POINT(SW/2.,0.,0.),(I-1)*60.,0.,0.),PKT(0)));
     END;
```

In ähnlicher Weise lassen sich auch komplexe Objekte beschreiben. Diese können dann je nach Anforderung der konstruktiven Aufgabenstellung zu Funktionskomplexen gruppiert und unter Beachtung der aktuellen Parameter dargestellt werden. Durch eine Hierarchie von Standardroutinen kann diese Zusammenfassung schichtweise auch Funktionsgruppen oder -hauptgruppen erfassen. Die Nützlichkeit dieses Vorgehens wird dadurch unterstrichen, daß nach einer Untersuchung im Werkzeugmaschinenbau /60/ mit nur 8 Funktionsgruppen etwa 70 % aller an Werkzeugmaschinen vorkommenden Elemente konstruiert werden können. Vorteilhaft ist dieses Vorgehen insbesondere auch bei Anpassungs- und Variantenkonstruktionen,deren absehbare Änderungen durch variable Parameter abgedeckt werden können.

Objektumfang

Im vorliegenden System wird ein Spektrum von Flächen bereitgestellt, aus dem sich ein umfangreicher Katalog von Standardkörpern spezifizieren läßt. Diese Auswahl kann dann, durch Nutzung der algorithmischen Fähigkeiten parametrisch variiert in einer komplexen Darstellung verwendet werden. Die Heranziehung des oben spezifizierten Standardobjektes 'Kopf einer Sechskantschraube' soll dies beispielhaft verdeutlichen (Abb. 5.1). Die Darstellung des Gesamtobjektes erfolgt durch den Befehl:

```
PLOT(COLL(SKS+SPACE(ROTATE(SHIFT(SKS,0.,0.,E),30.,0.,0.)
        +SWITCH(CYL(PKT(1),PKT(-1),E/6.)))+BOLZEN));
```

Die Mutter wird durch die Verschiebung und Rotation des Kopfes unter Hinzufügung der Bohrung gewonnen.

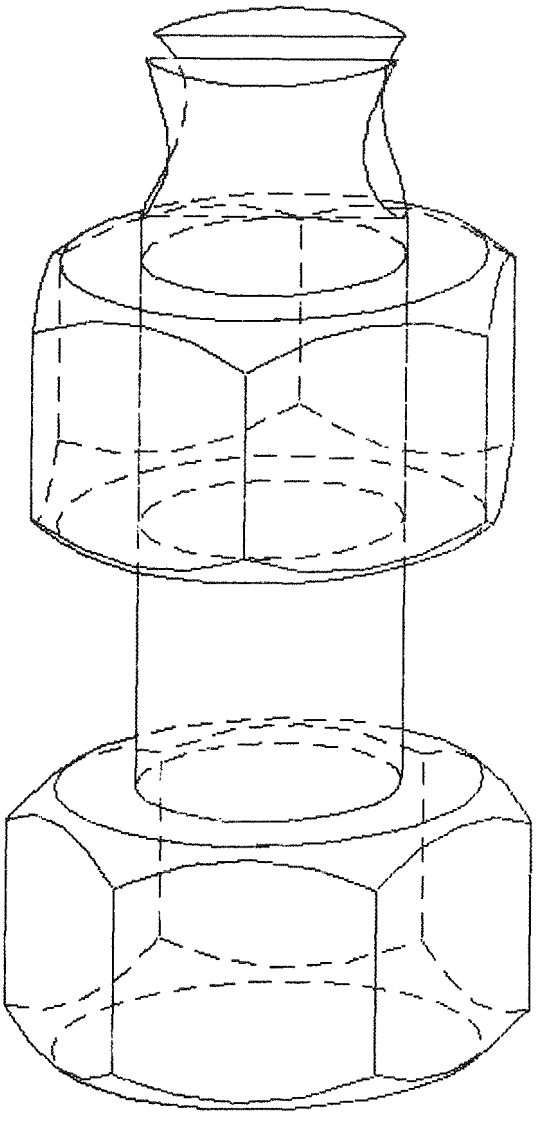


Abb. 5.1: Schraube mit Mutter erzeugt aus einer Grundform

Die Bereitstellung eines umfangreichen Vorrates an Standardobjekten ist nicht Aufgabe des Systemes, sondern seiner Anwender, die mit dem angebotenen Werkzeug ein weit über derzeitig verfügbare Fähigkeiten hinausgehendes Hilfsmittel zur Behandlung graphischer Probleme besitzen. Effektivitätsfördernd wirkt sich hierbei die Möglichkeit zur langfristigen Abspeicherung dieser Objektspezifikationen in übersetzter Form aus.

Hierzu sind die Beschreibungen häufig auftretender Körperformen als Modul abrufbar, in einer Bauteilbibliothek zu speichern. Bei dieser Vorgehensweise entfällt der Übersetzungsaufwand, da der ausführbare Code der Standardteilprogrammierung direkt in das Problemprogramm eingebaut werden kann.

Vervollständigt werden die graphischen Fähigkeiten durch die Vielfalt der graphischen Objekte zur Ausgabe der zu einer Zeichnung gehörenden organisatorischen Daten.

Modularität

Die Modularität ist bestimmend für die Wiederverwendbarkeit von Programmen /62/ und wird umso wichtiger, je größer die Komplexität der Funktionsträger und der Freiheitsgrad der Konstruktionsart ist.

Die Erzielung einer hohen Modularität war einer der Hauptgesichtspunkte bei der Konzeption des integrierten Systems REGENT und seiner Subsysteme. Bei der Implementierung von GIPSY wurden keine produkt- oder darstellungsspezifischen Einschränkungen zugelassen, um eine Anwendung für ein weites Aufgabenspektrum zu erlauben. Vom System her sind damit alle Voraussetzungen für einen modularen Aufbau von Darstellungs- und Auslegungsprogrammen gegeben.

Für den Anwendungsprogrammierer stellt sich die Aufgabe, diese Flexibilität bei der Programmierung durch eine hierarchische Gliederung der Implementierungs- und Funktionsebenen zu bewahren.

5.1.2 Darstellungshilfen

Bei der Erstellung technischer Abbildungen wird eine Vielzahl von Darstellungshilfen verwendet. Sie dienen der Verdeutlichung der graphischen Aussage. Die wichtigsten von ihnen sind:

- Zeichnen von Ausschnitten,
- Schnittdarstellung mit Schraffur,
- Explosionsdarstellung,
- Verschiedenheit von Projektionen (Zentral- und Parallel-Projektion),
- Standardansichten (Normalrisse, Di- und Isometrien),
- Weglassen oder Stricheln unsichtbarer Kanten,
- Verwendung von Sinnbildern,
- Ausgabe von Kurven in Diagrammen, usw..

Alle diese Techniken streben entweder durch ihre Nähe zur Realität oder die Einhaltung vertrauter Normen eine zweifelsfreie Informationsübermittlung an.

Im vorliegenden System sind, wie die beiden vorhergehenden Abschnitte zeigten, alle die obengenannten Fähigkeiten auch bei einer rechnerunterstützten Zeichnungserstellung verfügbar.

Die Anwendung von Darstellungen mit höherem Abstraktionsgrad wie z.B. /87/ würden den Rechnereinsatz nur begünstigen.

5.1.3 Informationsträger

Neben einer logischen Gliederung der graphischen Informationsträger /10/ nach

- Symbol- und Prinzipskizzen,
- Entwurfs-, Einzelteil- und Zusammenstellungszeichnungen,
- Bearbeitungs- und Montageplänen,
- Blockschaltbildern,
- Diagrammen, usw.

muß bei maschineller Zeichnungserstellung auch eine Einteilung nach physischen Gesichtspunkten getroffen werden. Wichtigste Medien für die Aufnahme sind hierbei

- Papier (Folien),
- Film und
- Bildschirm.

Da die Ansteuerung der für diese Medien erforderlichen unterschiedlichen Hardware nach dem heutigen Stand der Technik nur mit verschiedenen Softwarepaketen möglich ist, stellen sich bei jedem Übergang von einer Ausgabeform zur anderen Kompatibilitätsprobleme.

In REGENT wurde diese geräteabhängige Schnittstelle in den Systemkern verlegt und auf ein Minimum an Interfaceroutinen beschränkt. Der Anwender des Systems GIPSY wählt die von ihm gewünschte graphische Ausgabeeinheit nur durch die Angabe des Gerätetypes aus.

Derzeit wird die Benutzung folgender Geräte unterstützt

- Trommelplotter (CALCOMP)
- elektrostatischer Plotter (STATOS) und
- Speicherröhre (TEKTRONIX 4014).

Einer Ausweitung dieser Geräteauswahl zur Erzeugung farbiger Plots /82/ oder von Filmen /19,66,68/ zur Behandlung von Animation-Problemen /89, 153, 172/ steht nichts im Wege.

5.1.4 Zeichentechnik

Durch den Einsatz des Rechners ergibt sich eine der Betriebsweise entsprechende Zuordnung der Zeichentechniken nach:

- Stapelbetrieb und
- interaktiver Anwendung.

5.1.4.1 Ausführung im Stapelbetrieb

Im Falle eines Stapelauftrages werden alle bei der Bildgenerierung möglichen Varianten durch den Algorithmus des Programmes festgelegt. Die letztliche Auswahl erfolgt durch Eingabedaten und/oder logische Abfragen. Dieses Vorgehen wird auch als 'geschlossene' Programmierung bezeichnet /62,159/.

Obwohl sich die übliche Ausführung eines GIPSY-Programmes im Stapelbetrieb vollzieht, besitzt der Anwender die Möglichkeit, auch strukturelle Änderungen an seinen Zeichnungen vorzunehmen, da er mit dem System nicht nur über wenige ausgewählte Parameter kommuniziert, sondern über eine problemorien-

tierte Sprache, die ihn durch einen großen dem graphischen Problem angemessenen Objektvorrat unterstützt. Hierzu gehört auch die Komposition einer Abbildung höherer Komplexität aus bereits abgespeicherten, standardisierten Darstellungen von Funktionselementen.

5.1.4.2 Konzepte der interaktiven Anwendung

Für die graphische Datenverarbeitung bedeutet die Möglichkeit zur Interaktion die Erschließung der zweiten Dimension, wenn es sich bei dem graphischen Terminal um einen aktiven Bildschirm handelt.

Prinzipiell bieten sich für die Realisation eines interaktiv arbeitenden Systems unter Verwendung von GIPSY folgende Konzepte an:

- interaktive Programmentwicklung und
- Implementation eines Terminalsystems unter Nutzung der graphischen Fähigkeiten von GIPSY.

Interaktive Programmentwicklung

Voraussetzung für die interaktive Entwicklung von graphischen Problemprogrammen ist die Verfügbarkeit eines Dialogsystems. Die Timesharing Option (TSO) des Betriebssystems OS/360 erfüllt diese Bedingung. Ohne weitere Unterstützung wird jedoch hierdurch nur die Zusammenstellung und der Änderungsdienst von Programmen an ein Terminal verlegt. Die Ausführung des Programmes erfolgt weiterhin nach Übertragung des Auftrages ('Submit') im Stapelbetrieb. Es besteht daher keine Möglichkeit, nach der Ausführung von Teilschritten den Ablauf des Programmes korrigierend zu beeinflussen.

Durch die Heranziehung des Problemsprachenübersetzes (PLS) des REGENT-Systems ist auch die unabhängige Vorübersetzung der Problemstatements im Vordergrund des TSO möglich. Der Ablauf eines graphischen Programmes vollzieht sich auch unter TSO durch die Abfolge der Phasen:

- Vorübersetzung mit PLS,
- Übersetzung durch den PL/1-Compiler sowie
- Binden und Ausführen.

Der Anwender besitzt jedoch durch die Auflösung des Prozesses in diese Phasen die Möglichkeit nach jedem Teilschritt, das Ergebnis der bisherigen Ausführung zu bewerten und korrigierend einzugreifen. Diese Art der Pro-

grammentwicklung führt, wenn die Antwortzeiten der Rechenanlage auf eine Benutzeranforderung klein sind, schnell zu verfügbaren Ergebnissen.

Hat ein Programm alle Stadien der Übersetzung erfolgreich durchlaufen, so kann die Ausführung des Programmes und damit die Ausgabe der graphischen Daten auf dem Bildschirm erfolgen. Im vorliegenden System ist dies eine TEKTRONIX-Speicherröhre vom Typ 4014.

Implementierung eines interaktiven Terminalsystems

Im vorgenannten Fall beschränkt sich die Einflußnahme auf eine Programmausführung des im Vordergrund einer Timesharing-Umgebung ablaufenden Programmes auf die Spezifikation weniger Parameter für ein in seiner Struktur weitgehend festliegendes Objekt.

Ersetzt man dieses spezielle Programm durch eine Routine, die dem Anwender nicht nur die Angabe von Parametern, sondern die Spezifikation der Struktur von Objekten ermöglicht, so hat man ein interaktives graphisches System.

Die Erstellung eines derartigen Terminalsystems ist unter Anwendung der GIPSY-Fähigkeiten denkbar einfach, da nicht mehr die graphischen Objekte und Operationen selbst, sondern nur noch der Zugriff auf sie realisiert werden muß. So war mit GIPSY die Implementierung eines funktionsfähigen interaktiven Systems in nur drei Tagen möglich.

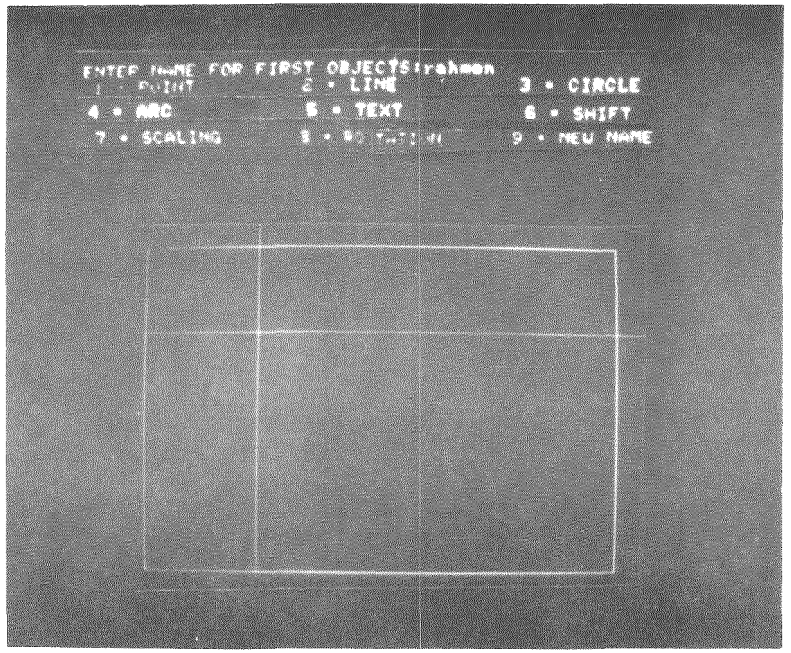
Die Fähigkeiten des Systems beschränken sich auf die Behandlung zweidimensionaler Objekte. Die graphischen Objekte werden mit einem Namen versehen und nach Eingabe ihrer charakteristischen geometrischen Größen mit dem Fadenkreuz in einer Listenstruktur abgespeichert. Zur Manipulation stehen die Lineartransformationen bereit.

Dem Benutzer wird dabei zu Beginn der Sitzung eine Auswahl der verfügbaren Fähigkeiten angeboten (Menü-Technik). Diese Optionsliste enthält neben

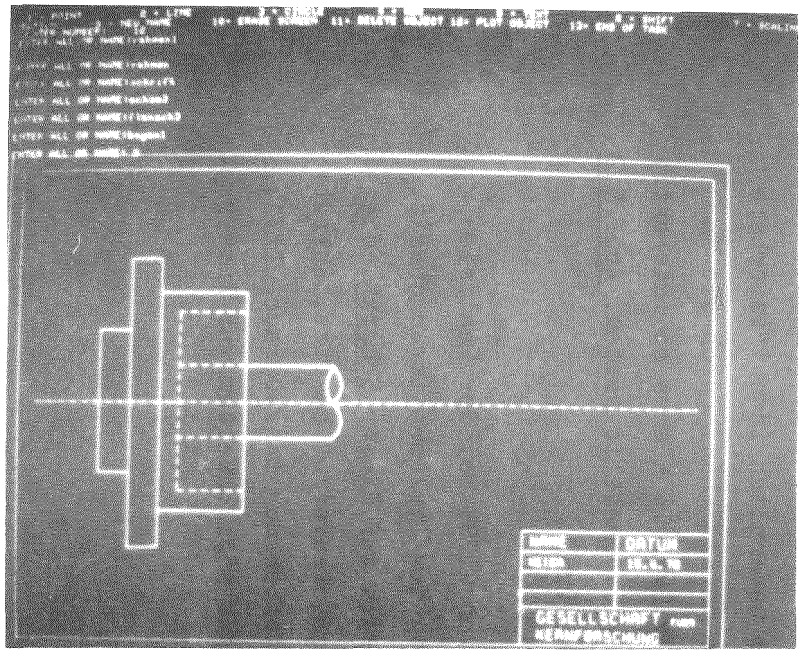
- den graphischen Objekten und Operationen
- die Möglichkeit zur Namensfestlegung und
- die Fähigkeit zur Eingabe von Koordinatenwerten über ein Fadenkreuz.

Es lassen sich auf diese Weise am Bildschirm namentlich identifizierbare Objekte erzeugen und graphischen Operationen unterziehen. Durch die Anwendung vorher erzeugter Standardobjekte oder Symbole können so interaktiv komplette Schaltbilder oder Zeichnungen effektiv erstellt werden. Abb. 5.2 zeigt einige Schritte einer Sitzung am Bildschirm.

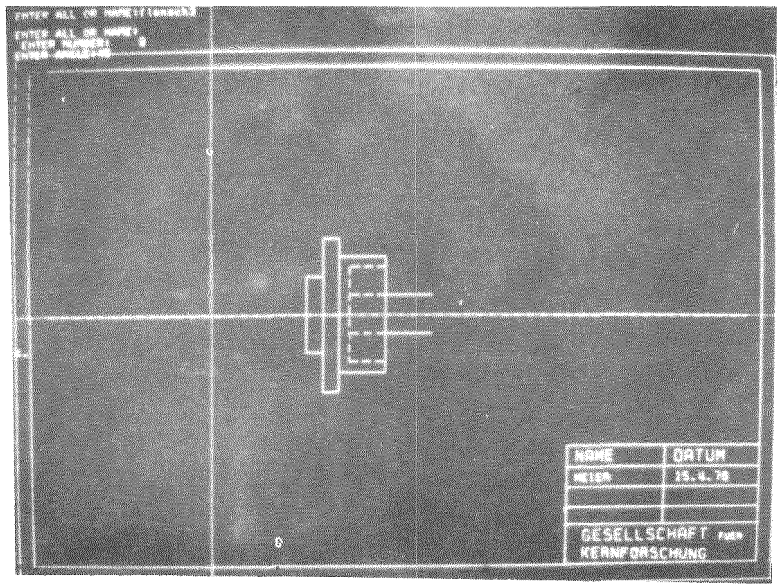
Der entscheidene Vorteil bei der Anwendung von GIPSY bei der Realisierung derartiger Systeme liegt in der Anhebung der Implementierungsbasis. Die Fähigkeiten des Terminalsystems können sich auf dieser Schicht, die den gewünschten graphischen Objekt- und Operatorenvorrat besitzt, abstützen. Verfügbar bleiben außerdem die Eigenschaften der Basissprache PL/1, von denen in diesem Zusammenhang vor allem die Möglichkeiten zur Interrupt-Behandlung zu nennen sind.



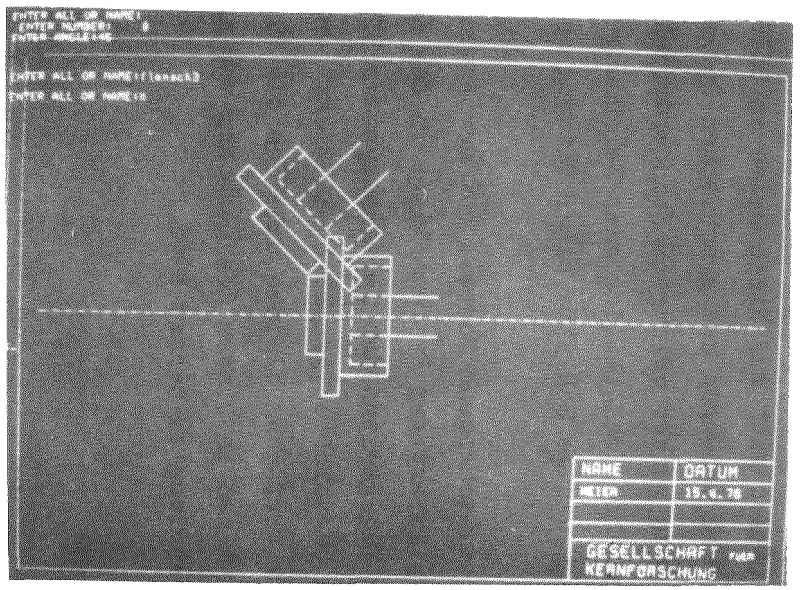
Menü und Fadenkreuz



Abrui eines Standardteils



Eingabe des Mittelpunktes und des Winkels (45°)
für eine Rotation (8) des Objektes (FLANSCH3)



Ausgabe nach erfolgter Rotation

Abb. 5.2: Interaktives Arbeiten am Bildschirm

5.2 Datenübertragung zwischen fachspezifischen Programmteilen

Für das Suchen und den Zugriff auf für den Entwurf relevante Daten benötigt der Konstrukteur einen wesentlichen Teil seiner Arbeitszeit /23/. Die Möglichkeiten zur Bereitstellung dieser Informationen werden in /13/ und /157/ diskutiert. Der Rechner kann dabei durch die Beschleunigung der mit hohem organisatorischen Aufwand zur Erfassung, Aufbereitung und Rückgewinnung verbundenen Tätigkeiten für einen besseren Informationsfluß sorgen.

Im System REGENT, das eine koordinierte Ausführung von Aufgaben aus verschiedenen Problembereichen vorsieht, tritt die Notwendigkeit für den Austausch von Daten nicht nur zwischen dem Anwender und dem Rechner, sondern vor allem auch für die Kommunikation zwischen Subsystemen auf.

Komplexe Entwurfsaufgaben erfordern zu ihrer Lösung die Fähigkeiten verschiedenster Fachbereiche. Im Sinne einer effektiven Bearbeitung sollten die das Problem beschreibenden Daten nur einmal für alle Auslegungsphasen übergeben werden müssen. Dies läßt sich nur erreichen, wenn ein Datentransfer zwischen den fachspezifischen Subsystemen möglich ist.

In REGENT wird die Datenübergabe durch

- interne (globale Variable, Parameter, usw.) oder
- externe (Dateien, Datenbanken)

Kommunikationshilfen unterstützt (vgl. Abschnitt 3.4).

Dieser Austausch erstreckt sich neben den modellbeschreibenden Eingabedaten auch auf die Ergebnisse vorangegangener Auslegungsrechnungen.

Durch diese Transparenz wird es ermöglicht, systemweit auf wichtige Daten zuzugreifen, ohne eine manuelle Aufbereitung von Ergebnissen für den nächsten Auslegungsschritt zu benötigen. Diese Fähigkeit ist eine wichtige Voraussetzung für Beibehaltung hoher Modularität bei integrierten Systemen, da trotz unabhängiger Programmierung die gemeinsame Nutzung von Moduln z.B. für strukturmechanische, fluiddynamische oder werkstofftechnische Aufgaben erfolgen kann /145,149,152/.

Durch den Zugriff auf die aus diesen Rechnungen folgenden geometrischen Abmessungen ist eine Zeichnungserstellung durch den Einsatz der graphischen Fähigkeiten des Subsystems GIPSY ankoppelbar.

Beispielhaft soll dies die Darstellung eines Reaktor-Containments des in REGENT implementierten Subsystems zur Erzeugung von Netzwerken demonstrieren (Abb. 5.3). Die graphische Ausgabe erfolgt ausschließlich unter Verwendung von GIPSY Statements und basiert auf den Daten des aufrufenden Systems.

Es war deshalb im vorliegenden Fall nicht erforderlich, die umfangreiche Datenmenge, die die Containment-Struktur beschreibt, zu kopieren oder auf einem externen Medium zwischenzuspeichern. Durch den geschachtelten Aufruf von Subsystem (vgl. Abschnitt 3.4) sind alle Daten des die graphischen Fähigkeiten anfordernden Subsystems verfügbar.

Durch die graphische Ausgabe wird die Beseitigung fehlerhafter Koordinatenangaben von Knotenpunkten erleichtert und beschleunigt.

Bei einer anderen Anwendung des Systems setzt die graphische Ausgabe nicht erst zum Abschluß des Konstruktionsprozeß, sondern bereits in der Entwurfsphase ein. Die Anordnung der Komponenten einer verfahrenstechnischen Anlage wird für den Konstrukteur aus einer perspektivischen Ansicht ersichtlich. Diese Darstellung verschafft einen Überblick über die Gesamtanlage und erleichtert die Entscheidungen zur Anbringung von Durchführungs- oder Verbindungskanälen. (Abb. 5.4)

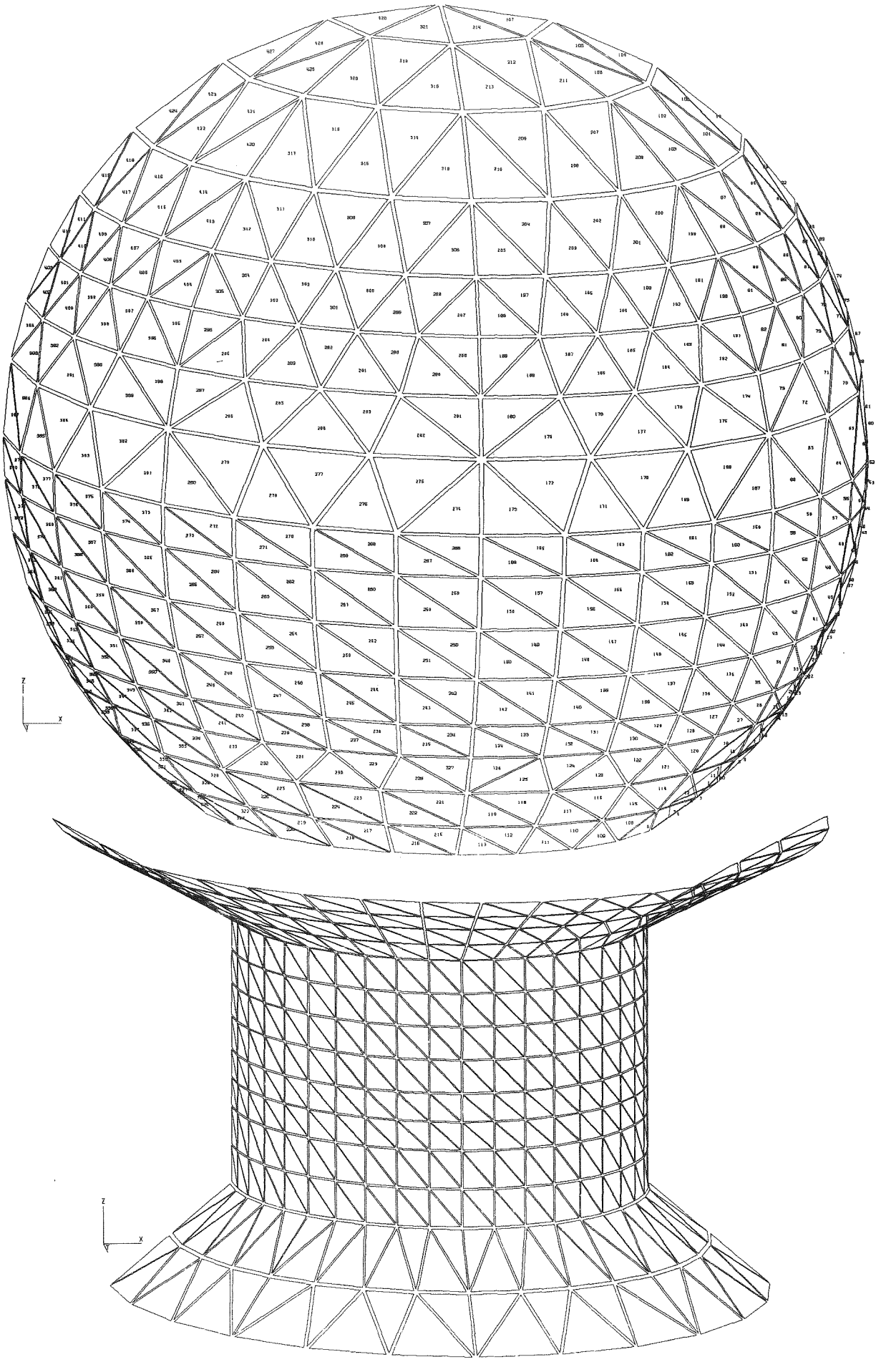


Abb. 5.3: Darstellung der Finitelement-Struktur eines Reaktor-Containments

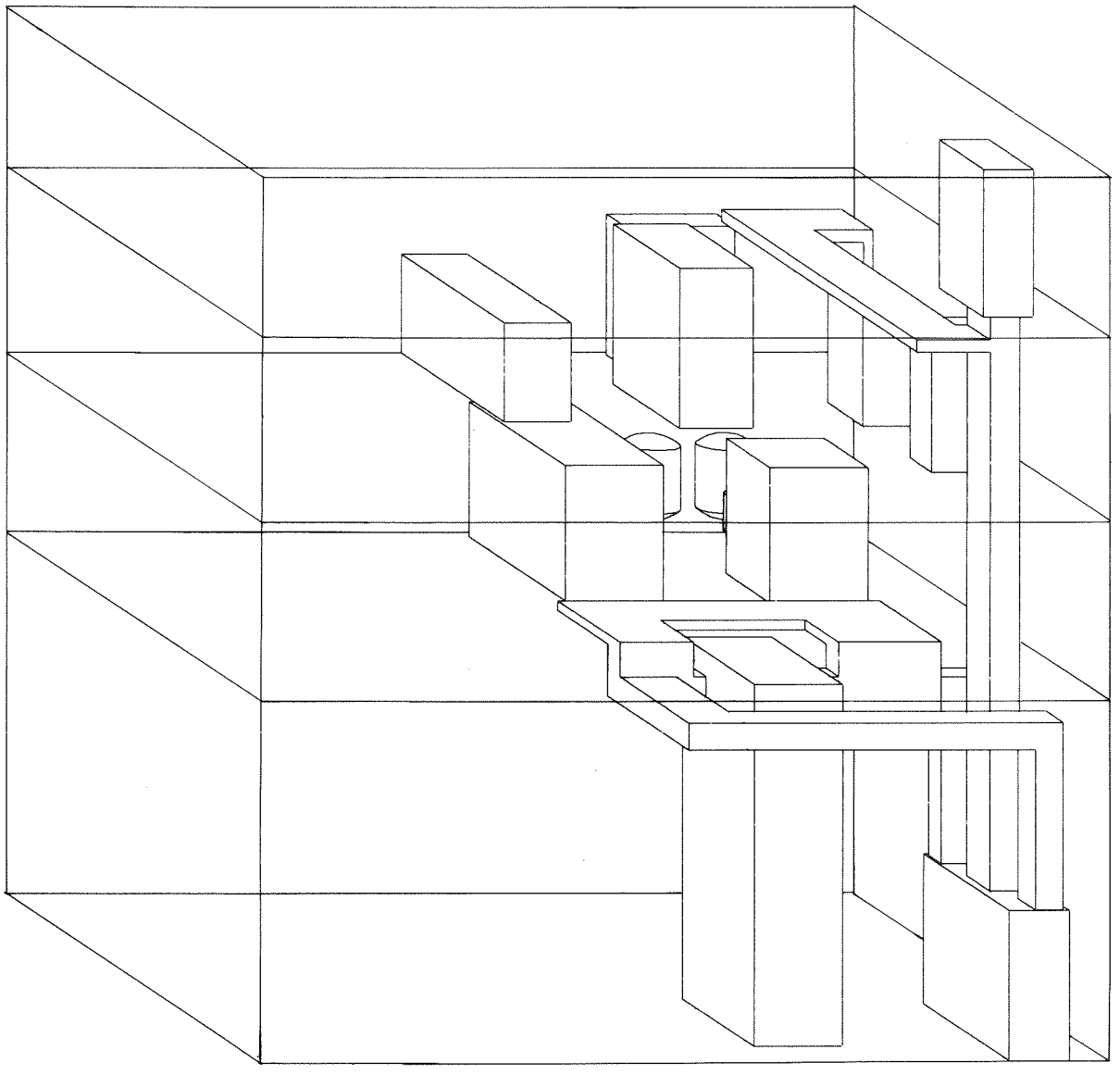


Abb. 5.4: Anordnung von Anlagenkomponenten

6. Effektivitätsbetrachtungen

Die Effektivität eines Systems läßt sich nach den Gesichtspunkten:

- Speicherplatzbedarf und
- Rechenzeit

bewerten. Die für diese Kriterien geltenden Zahlen lassen sich leichter relativieren als Angabe zu den Kosten für eine Programmausführung, da diese durch die spezielle Charakteristik von Abrechnungsalgorithmen beeinflußt sind.

Die im folgenden angegebenen Arbeitsspeicheranforderungen und Ausführungszeiten beziehen sich auf eine Rechenanlage des Typs IBM 370/168 mit dem Betriebssystem OS/MVT und werden in Kilo-Bytes (1 KB=1024 Bytes mit je 8 Bits) bzw. als CPU-Zeit in Sekunden gemessen.

Speicherplatzbedarf

Die Übersetzung der GIPSY-Sprache erfolgt unter Verwendung des Problemsprachensystem PLS und des IBM-PL/1-Optimising Compilers, die beide für eine effektive Ausführung etwa 200 KB Speicher erfordern. Bei kleinerem Arbeitsspeicherangebot geht die Performance merklich zurück.

Der Speicherbedarf bei der Ausführung der graphischen Befehle gliedert sich in den Platz für:

- die Routinen und
- die Daten.

Durch die Einbettung von GIPSY in ein integriertes System kann der Vorteil des dynamischen Aufrufens von Modulen genutzt werden, der Anspruch an Speicherplatz läßt sich damit auf die jeweils erforderlichen Routinen reduzieren. Im Hinblick auf eine effektive Behandlung sollte aber auch hier der Kernspeicher nicht auf das zur Berechnung unabdingbare Minimum beschränkt werden, da dann die Rechenzeit durch ständige Auslagerungsvorgänge ansteigt. Der Speicherplatzbedarf für die Ausführung eines graphischen Programmes setzt sich wie folgt zusammen:

- Hilfsroutinen der Basissprache	28 KB
- Routinen des Systemkerns	16 KB
- Plotterinterface im Systemkern	5 KB

- Statischer Programmbereich des Subsystems	34 KB
- Dynamischer Programmbereich des Subsystems bei Strichgraphik/Körperbehandlung	40/100 KB
- Programme für die Plotteransteuerung und Interfaceroutinen dazu	50 KB
- Graphische Programme für die aktuelle Anwendung (je nach Problem-Umfang)	10(-30) KB

Diesem Speicheraufwand für die Module sind etwa 24 KB für den dynamischen Datenbereich hinzuzurechnen. Da das Verfahren zur Berechnung und Ausgabe von Körperkanten nicht die gleichzeitige Abspeicherung mehrerer Raumkurven erfordert und damit der Speicherplatz für Kurvendaten weitgehend unabhängig von der Problemgröße ist, reicht die obige Zuteilungsgröße für den Datenbereich in aller Regel aus.

Nachteilig macht sich das Fehlen einer in PL/1 oder in ASSEMBLER für den Aufruf aus PL/1 geschriebenen Plottersoftware auf die Arbeitsspeicheranforderung bemerkbar, das einen Interface-Aufwand von etwa 35 KB für die Überwindung der Sprachgrenzen zwischen PL/1 und FORTRAN bedingt.

Im allgemeinen ist für die Ausführung eines graphischen Programmes eine Kernspeicherzuteilung von 200 KB für Strichobjekte und 250 KB für die Darstellung von Körpern ausreichend.

Rechenzeit

Die Gesamtrechenzeit zur Lösung eines graphischen Problems ist die Summe aus folgenden Teilschritten:

- Übersetzung,
- Strukturaufbau,
- Kurvenberechnung und
- Sichtbarkeitstest.

Der Übersetzungsvorgang beinhaltet die Analyse der POL-Statements mit Hilfe des PLS-Systems und die anschließende Generation von Maschinencode durch den Compiler. Da beide Übersetzer etwa 80 Anweisungen/sec verarbeiten /45/, ergibt sich meist eine Rechenzeit für beide Teilschritte zusammen, die unter 10 sec liegt. Hierbei ist der Compilationsanteil meist etwas höher als der der Vorübersetzung, da das POL-Programm bei der Umwandlung in gültiges PL/1 eine starke Expansion erfährt.

Der Strukturaufbau und die Behandlung graphischer Strichinformation bedingen einen geringen zusätzlichen Rechenaufwand. Ein Vergleich des Aufwandes für die Nutzung dieser Fähigkeiten zwischen GIPSY, dem ICES-Subsystem GRAPHIC /47/ und einem direkten Aufruf der Basissoftware eines Plotterherstellers /33/ soll anhand einer praktischen Anwendung durchgeführt werden.

Die Aufgabe bestand darin, Ätzvorlagen für die Leiterbahndurchführungen einer Funkenkammer zu erstellen. Abb. 6.1 zeigt die Ausgabe eines Stranges und eine Gegenüberstellung des erforderlichen FORTRAN- und GIPSY-Programmes.

Der Aufwand zur Codierung und Ausführung der verschiedenen Versionen wurde dokumentiert.

		GRAPHIC	FORTRAN (CALCOMP)	GIPSY
Aufwand	CPU-Zeit /sec/	249	8.8	8.54(3.86)
	Programmierung /min/	30	120	25
Auffinden von Programmfehlern durch	unabhängigen Syntaxcheck	-	möglich	möglich
	Testhilfen	befriedigend	schlecht	gut

Die interpretative Abarbeitung der Problemstatements bei GRAPHIC führt zu etwa 30mal höheren Ausführungszeiten als bei beim compilierenden Vorgehen. Für eine wiederholte Anwendung des Programmes kann bei GIPSY zudem von einer übersetzten und in ausführbarer Form gespeicherten Version ausgegangen werden, wodurch sich die Rechenzeit auf 3.86 sec und damit auf etwa ein Sechszigstel gegenüber GRAPHIC verringern ließe.

Die Ausführungszeiten des FORTRAN-Programmes und der GIPSY-Anwendung unterscheiden sich nur unwesentlich. Erfaßt man jedoch den Aufwand zur Erstellung und Austestung der Routinen mit, so zeigt sich deutlich der Vorteil der problemorientierten Sprache. Die Codierung des Problems benö-

tigte mit GIPSY nur etwa ein Fünftel der Zeit, die FORTRAN beanspruchte. Die Fehlerfreiheit des Programmes war durch die guten Testhilfen des PLS-Vorübersetzes sowie des PL/1-Compilers und-Laufzeitsystems schnell gesichert.

Die Kurvenberechnung erfolgt nach analytischer Berechnung der bestimmenden Koeffizienten der allgemeinen Gleichungsformen allein durch inkrementale Änderung eines Parameters. Die Effektivität eines solchen Verfahrens ist durch keinen numerischen Lösungsversuch zu übertreffen. Durch eine günstige Wahl der Inkremente läßt sich für eine wiederholte Ausgabe eine objektspezifische Minimierung der Rechenzeiten erzielen.

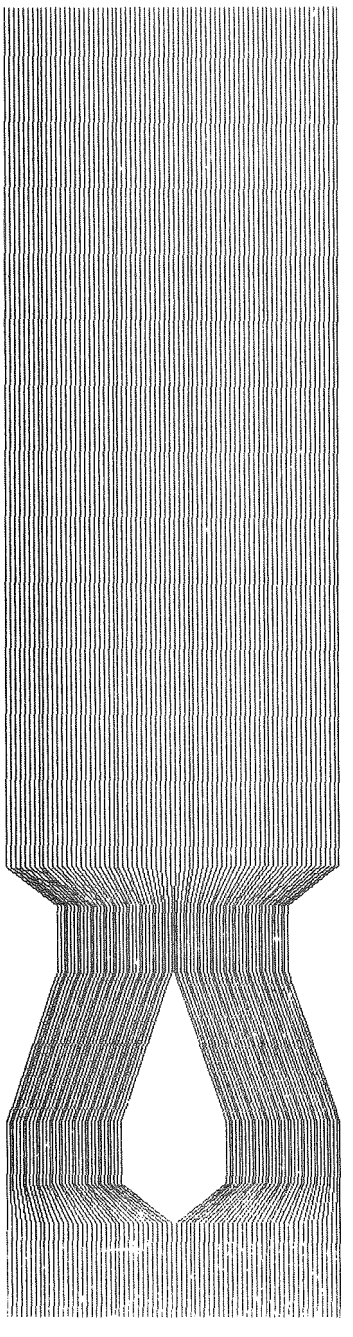
Die Rechenzeiten für die Entscheidung der Sichtbarkeit von Kurvenpunkten bestimmen in hohem Maße die Effektivität eines Systemes. Obwohl in der Literatur ausführlich die Eigenschaften von Visibilitätsverfahren diskutiert werden (vgl. Abschnitt 2.2 und 2.3), gibt es nur wenige quantitative Angaben zum erforderlichen Rechenaufwand. Die Arbeiten von Herold/71/ und Woon /164/ enthalten Daten über die Effektivität der dort implementierten Verfahren. Sie sind jedoch beide für einen Vergleich nicht geeignet, da es sich um Algorithmen handelt, die ausschließlich entweder mit Polyedern oder mit Quadriken arbeiten und aus der Einschränkung auf die jeweilige Flächenform Vorteile ziehen.

Für die Ausführungszeiten von in dieser Arbeit abgebildeten Objekten wurden folgende Werte gemessen:

Objekt	Abb.	Flächen	CPU - Zeit [sec]		mit / ohne
			ohne Visib.-Test	mit Visib.-Test	
Kegel	4.4	4	8.1	12.5	1.52
Würfel (alle 8)	4.9	7	35.6	76	2.13
Sockel	6.3	24	16.3	40	2.45
Schraube	5.1	25	15	52	3.47
FR2	6.2	40	39	96	2.46

FORTRAN

```
      DIMENSION IBUF(2000)
      REAL*4 X1(1000),Y1(1000)
      REAL*8 X(8),Y(8),DIF,XDIF,YDIF,R,XINC,YINC
      CALL CALCIN(IBUF,2000,0)
      FAKT1=1.D0/25.4D0
      CALL FACTOR(FAKT1)
      CALL PLOTC(100.,0.,-3)
      DIF=124.D0/74.D0
      Y(1)=0.
      Y(2)=500.
      Y(3)=547.
      Y(4)=563.
      Y(5)=637.
      Y(6)=653.
      Y(7)=675.
      Y(8)=725.
      X(5)=25.D0
      X(6)=25.D0
      X(3)=1.D0
      X(4)=1.D0
      DO 1 I=1,150
      X(1)=2*I-1
      X(2)=2*I-1
      X(7)=2*I-1
      X(8)=2*I-1
      WRITE(6,1000) I,(X(KK),KK=1,8)
1000  FORMAT(' *I*',I3,'*X1*',F8.3,'*X2*',F8.3,'*X3*',F8.3,'*X4*',F8.3,'
     **X5*',F8.3,'*X6*',F8.3,'*X7*',F8.3,'*X8*',F8.3)
      XXXX=X(1)
      YYYY=Y(1)
      CALL PLOTC(XXXX,YYYY,3)
      NPUN=1
      DO 30 K=2,8
      XDIF=X(K)-X(K-1)
      YDIF=Y(K)-Y(K-1)
      R=DSQRT(XDIF**2+YDIF**2)
      M=R
      M=M+1
      XINC=XDIF/M
      YINC=YDIF/M
      DO 40 MLOOP=1,M
      X1(NPUN)=X(K-1)+(MLOOP-1)*XINC
      Y1(NPUN)=Y(K-1)+(MLOOP-1)*YINC
40    NPUN=NPUN+1
30     CONTINUE
      X1(NPUN)=X(8)
      Y1(NPUN)=Y(8)
      X1(NPUN+1)=1.
      Y1(NPUN+1)=1.
      X1(NPUN+2)=1.
      Y1(NPUN+2)=1.
      CALL LINE(X1,Y1,NPUN,1,0,0)
      X(3)=X(3)+DIF
      X(4)=X(4)+DIF
      X(5)=X(5)+DIF
      X(6)=X(6)+DIF
      IF (I.NE.75) GO TO 1
      X(3)=175.D0
      X(4)=175.D0
      X(5)=   151.DC
      X(6)=151.D0
1      CONTINUE
      CALL PLOTC(X1(1)+50.,Y1(1),999)
      STOP
      END
```



GIPSY

```
BAHNEN:PROC OPTIONS(MAIN) REGENT(PLOT=STATOS,NOCA);
/*****************************************************************/
ENTER GIPSY;
   DCL P(9) POINT2;
   DCL (XR,
        X(8) INIT((2).5,(2)11.5,(4).5),
        Y(8) INIT(86.5,73.,41.5,28.5,20.,C.,275.,95.))
             DEC FLOAT(6);
   DO XR=1.,-1.;
      DO I=1 TO 8;
         SET P(I)=POINT2(35.5+X(I)*XR,Y(I));
      END;
      DO J=1 TO 35;
         DO I=1 TO 8;
            IF I<5 THEN SET P(I)=SHIFT2(P(I),XR*.6944,0.);
            ELSE SET P(I)=SHIFT2(P(I),XR,0.);
         END;
         SET P(9)=P(1);
         DO I=1 TO 5,7,8;
            PLOT(LINE(P(I),P(I+1)));
         END;
      END;
   END;
END GIPSY;
/*****************************************************************/
FINISH;
END BAHNEN;
```

Abb. 6.1: Programmvergleich für die Erstellung von Ätzvorlagen für Leiterbahndurchführungen

Neben der Anzahl der Flächen, der Form der begrenzenden Flächen und damit der Schnittverläufe sind die oben bereits angesprochenen Inkremente Einflußfaktoren auf die Effektivität der Bildgenerierung. Für diesen Einfluß ergab sich für die zentralprojizierten Reaktorgebäude des FR 2 (Abb. 6.2) der folgende Zusammenhang:

Inkrement	CPU-Zeit [sec]
4	96
8	60
14	48

Die Wahl eines optimalen Inkrementes für das Entlanglaufen auf den Körperkanten ließe sich aus den räumlichen Abmessungen der Objekte automatisch vornehmen. Hinsichtlich der Erfordernisse eines Visibilitätstests kann diese Bestimmung jedoch nicht erfolgen, da diese Größe in nicht vorhersehbarer Weise von der Projektionsart- und richtung abhängt.

Bei der Wertung der obigen Ausführungszeiten ist zu berücksichtigen, daß durch die Eingliederung von GIPSY in das integrierte System REGENT und die damit verbundene Möglichkeit des dynamischen Linkens bei exzessivem Subroutinenaufruf - dies geschieht besonders beim Visibilitätstest - eine Effektivitätseinbuße zu verzeichnen ist. Eine quantitative Aussage über den Einfluß dieser Systemeigenschaft, die andererseits auch den Vorteil eines modularen und speichereffektiven Programmaufbaues bringt, konnte bisher nicht gewonnen werden.

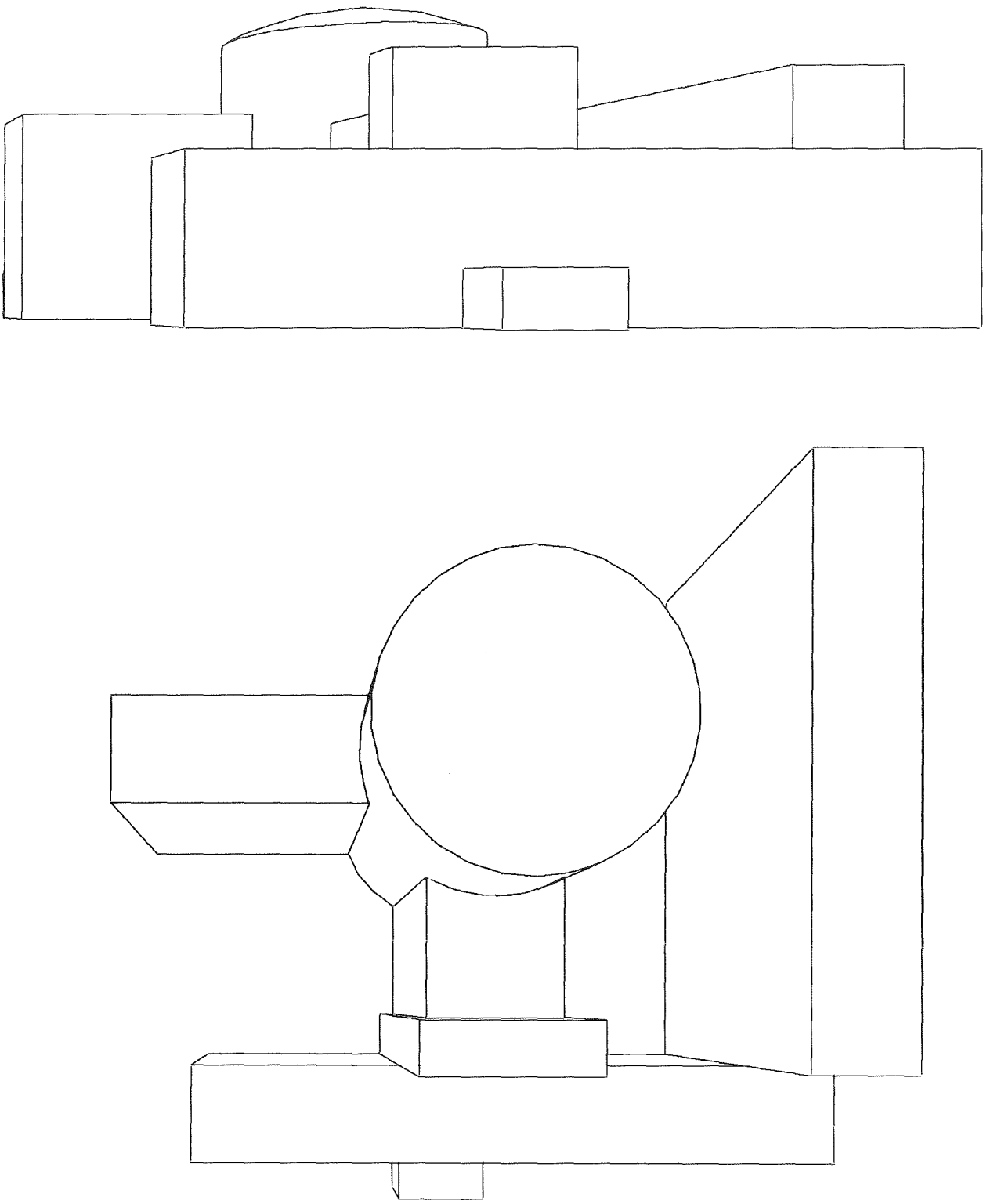


Abb. 6.2: Perspektivische Ansichten des FR 2-Gebäudekomplexes

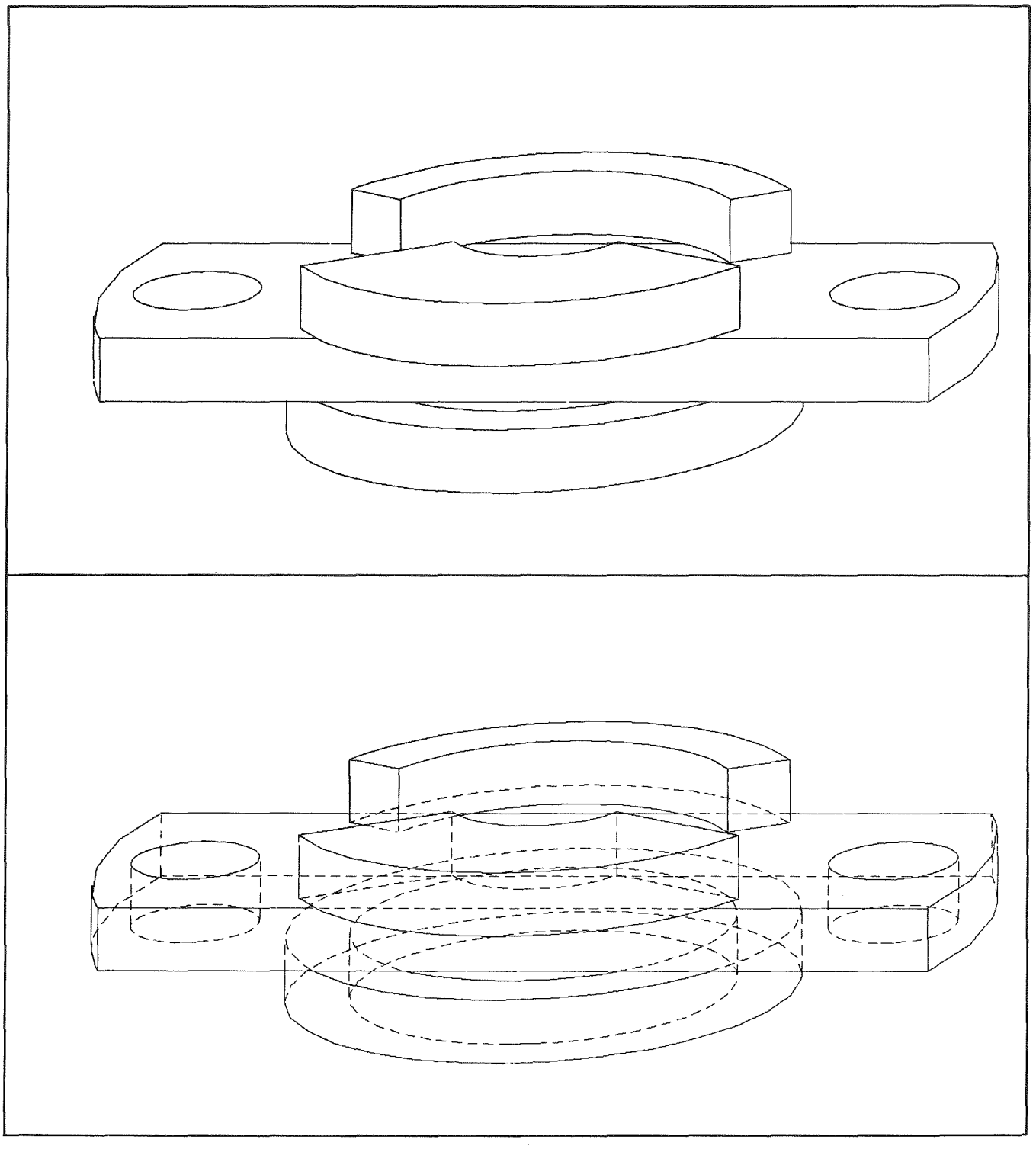


Abb. 6.3: Sockel

7. Zusammenfassung der wichtigsten Ergebnisse

Zur Lösung graphischer Aufgaben im System für das rechnergestützte Entwickeln und Konstruieren REGENT wurde das Subsystem GIPSY bereitgestellt.

Das Spektrum der behandelbaren graphischen Probleme umfaßt

- die 2D- und 3D-Strichgraphik,
- die Erstellung von Diagrammen und
- die Körperdarstellung mit Flächen 2. Ordnung unter Anwendung eines Sichtbarkeitskriteriums.

Die Formulierung der Aufgaben erfolgt in einer problemorientierten Sprache, die als Erweiterung von PL/1 realisiert wurde. In allen Problemprogrammen stehen daher die Fähigkeiten der Basissprache

- zur Vereinbarung von Prozeduren,
- zur Ablaufsteuerung,
- für die Behandlung alphanumerischer und logischer Größen,
- für die Ausführung von Ein/Ausgabeoperationen , usw.

bereit. Der Sprachumfang wurde durch eine große Auswahl an graphischen Datentypen erweitert. Die Behandlung dieser Daten kann nach den von PL/1 für Problemdaten vertrauten Regeln erfolgen:

- namentliche Identifikation von Objekten,
- Möglichkeit zur Deklaration von Aggregaten,
- Nutzung verschiedener Speicherklassen,
- flexible Schachtelung von Ausdrücken,
- Argumentübergabe an Prozeduren, usw..

Durch die Ausnutzung dieser Spracheigenschaften wird die Programmierung graphischer Aufgaben wesentlich erleichtert, da sich die dem Anwender in der Anschauung vertrauten Objekte im Sprachumfang wiederfinden. Für die Manipulation steht eine große Auswahl an graphischen Operationen, die die wichtigsten Zeichentätigkeiten nachvollziehen helfen, bereit. Alle Angaben erfolgen in einer einfachen problemangepaßten Syntax. Dennoch ist für die effektive Anwendung und die zweckmäßige Festlegung algorithmischer Ausgabebeschreibungen eine Beherrschung von PL/1 vorteilhaft.

Diese Spracheigenschaften gelten für alle graphischen Problembereiche. Neben der Behandlung von graphischer Strichinformation und der Darstellung von Wertefeldern in Diagrammen oder unter Zuhilfenahme perspektivischer Ansichten und Höhenlinien fand die Beschreibung von Körpern besondere Beachtung.

Die räumlichen Objekte - auch konkave - können durch

- plane,
- sphärische,
- zylindrische oder
- konische Flächen

begrenzt sein. Die Darstellung erfolgt durch

- Durchdringungs- und
- Umrißkurven,

die für beliebige Lage der Flächen zueinander bzw. bei beliebiger Projektion analytisch bestimmt und auf eine frei wählbare Bildebene abgebildet werden. Das Verfahren beruht auf einer kompakten internen Darstellung der Flächen durch die geometrisch unabdingbaren Angaben und benötigt keine Zwischenspeicherung der erzeugten Raumkurven, so daß der Speicheraufwand auf die gerade untersuchte Kurve beschränkt bleibt.

Die Spezifikation der Objekte erfolgt durch eine kompakte Eingabe unter Nutzung einer einfachen Körperarithmetik, wobei auch Lineartransformationen zugelassen sind. Prinzipiell gibt es keine Begrenzung für die Körperanzahl und deren begrenzende Flächen.

Zur Verdeutlichung der graphischen Ausgabe ist die Verwendung von

- Ausschnitten,
- Schnitten (Schraffuren) und
- Explosionsanordnungen

möglich.

Für alle möglichen Flächenkombinationen ist die Ausgabe unter Anwendung eines Visibilitätskriteriums möglich. Bei dem implementierten Verfahren handelt es sich um einen kombinierten Kanten- und Punktetest, der aufgrund analytisch vorbereiteter Berechnungen und vefahrensbeschleunigender Zusatzbedingungen entscheidet. Der Algorithmus ist auch auf von der Körperfläche unabhängige Raumpunkte anwendbar.

Diese Fähigkeiten zeichnen das vorliegende System als ein universell einsetzbares Hilfsmittel zur Erzeugung graphischer Darstellungen aus.

Die vorgestellten Methoden zur Körperbeschreibung erlauben die Behandlung von Werkstücken großer Variationsbreite. Die beschriebenen Formen lassen sich in Bibliotheken abspeichern und mit Parametervariation wieder abrufen. Durch eine hierarchische Gliederung von Standardroutinen ist so eine schichtweise Programmierung von Grundelementen über Funktionskomplexe zu Baugruppen möglich.

Das System unterstützt die Bearbeitung graphischer Aufgaben in einer dialogfähigen Umgebung.

Die analytischen Methoden zur Bestimmung der Körperkanten bedingen einen geringen Speicher- und Rechenaufwand. Die Übertragbarkeit dieser Fähigkeiten auf Kleinrechner ist dadurch gewährleistet.

Das vorliegende System ist kein "geschlossenes" Programm. Es kann vielmehr Grundlage für weitergehende Entwicklungen sein, da es aufgrund seiner Fähigkeiten generell eine Anhebung der Implementierungsbasis für graphische Problemlösungen bewirkt.

Durch die zur Zeit anlaufende allgemeine Anwendung des Systems werden sich Anregungen für fortführende Arbeiten ergeben, von denen sich bereits heute abzeichnen:

- Aufbau eines interaktiven graphischen Arbeitsplatzes,
- Speicherung der graphischen Objekte und deren graphischer Ausgabe in interpretierbaren Dateien,
- Bereitstellung von POL-Programmen für Standardaufgaben in Anwenderbibliotheken und
- der Einsatz von Kleinrechnern zur Unterstützung des Dialoges.

Seite

Anhang

Anhang

A. Syntax und Semantik der GIPSY-Sprache

A1. Verwendete Syntaxnotation

Die Metasprache zur Beschreibung der Syntax benutzt folgende Symbole:

< > Die spitze Klammer schließt variable Sprachelemente (non-terminals) ein, die für Benennungen, Konstanten oder Ausdrücke der Basissprache stehen oder als Platzhalter für genauer zu spezifiziertende Syntaxkonstruktionen der Spracherweiterung dienen.

::= Durch dieses Symbol werden die Regeln für die Generation einer gültigen Syntax durch die Substition der angegebenen linken durch die rechten Seiten festgelegt.

| Dieses Zeichen trennt alternative Erzeugungsvorschriften.

$[\ ]_n^m$ Alle Elemente in dieser Klammer können wiederholt auftreten, wobei n die minimale m die maximale Anzahl der Wiederholungen kennzeichnet. Fehlt die Angabe von n oder m, so wird eins angenommen. Steht anstelle von m ein Stern (*) und beliebige Wiederholungen erlaubt.

Wird weder m noch n angetroffen, so ist n = 0, d.h. die Elemente in der Klammer können auch wegfallen.

Sind innerhalb der Klammer mehrere Möglichkeiten der Formulierung untereinander aufgelistet, so sind dadurch ebenfalls verschiedene Alternativen beschrieben, von denen die erste die Standardannahme darstellt.

Großbuchstabige Texte sind konstante Elemente (terminals) der beschriebenen Anweisung, die bei einer Anwendung in dieser Form im codierten Statement erscheinen müssen.

Unterstreichungen bei großbuchstabigen Texten kennzeichnen die abgekürzte Form von Schlüsselwörtern der Sprache.

Bezüglich der nicht näher beschriebenen Anweisungen oder Sprachelemente gelten die Regeln von PL/1. Dies trifft insbesondere zu auf:

- die Regeln der Namensgebung,
- die Verwendung von Trennzeichen,
- den Aufbau von Konstanten und
- die Benutzung von Operatoren.

A2. Anweisungen

Die GIPSY-Sprache wurde auf der Basis von PL/1 entwickelt. Die Erzeugung, Behandlung und Ausgabe von graphischer Information erfolgt mit den Mitteln der Spracherweiterung in Abhängigkeit von geometrischen und/oder graphischen Argumenten, die als Ausdrücke ('expressions') spezifiziert werden.

Für Ausdrücke gelten die Regeln von PL/1, wonach diese gebildet werden aus:

- Konstanten,
- Variablen sowie
- Kombinationen von Konstanten und/oder
- Variablen mit Operatoren und Klammern.

Nach dem Ergebnis der Auflösung dieser Formen werden Element- und Feldausdrücke unterschieden und als < elementexpr> bzw. < arrayexpr> in der Syntaxbeschreibung verwendet.

Nach den Basisdatentypen von PL/1 bestimmen diese Ausdrücke arithmetische Werte oder Zeichenketten ('strings'). Letztere werden zur deutlicheren Unterscheidung als < stringexpr> gekennzeichnet.

Die bei der Spezifikation graphischer Daten verwendeten arithmetischen Werte betreffen meist Längen- oder Winkelangaben. Zur Erleichterung der Handhabung dieser Größen wird deren Verwendung in einheitenbehafteter Form unterstützt.

Für Längenangaben gilt dabei:

< lvalue> : := < elementexpr> < lunit>

mit

< lunit> : := [MM | CM | M | INCH | FOOT | YARD]

Die Eingabe von Winkelgrößen erfolgt durch:

< avalue> : : = < elementexpr> < aunit>

und

< aunit> : : = [DEGREE | RADIAN]

A2.1 Deklaration der graphischen Daten

Für alle Variablen, deren gewünschter Datentyp sich nicht aus dem Kontext des Programmes oder einer Standardannahme ergibt, muß in PL/1 eine Deklaration der Attribute erfolgen. Für graphische Daten ist eine Vereinbarung des Datentypes immer erforderlich und ist wie folgt aufgebaut:

DECLARE [< level>] < ident> [< dim>] < type> [< length>] [storage class];

Eine graphische Problemvariable, deren Namen < ident> nach normalen PL/1-Konventionen gebildet wird, erfährt durch diese Anweisung die Festlegung ihres Attributsatzes.

Für die einzelnen Attribute gilt:

< level>	Bestimmung des Levels innerhalb von Datenaggregaten.
< dim>	Vereinbarung einer graphischen Feldvariablen durch Angabe der Dimension und der Feldgrenzen. Es gelten die PL/1-Regeln, d.h. es ist z.B. erlaubt, die Feldgrenzen durch Ausdrücke zu spezifizieren und Indizes für negative Werte vorzusehen.
< type> ::=	Zur Behandlung der graphischen Aufgaben wurde die Sprache um folgende Datentypen, deren Deklaration mit den Basistypen gemischt erfolgen kann, erweitert:
POINT|	3D-Punkt,
TEXT|	3D-Text,
POLYGON|	3D-Polygonzug,
CIRCLE|	3D-Kreis,
PLANE|	Ebene,
BALL|	Kugel,
CYLINDER|	Zylinder,
CONE|	Kegel,
SPACE|	Raumelement,
COLLECTION|	Kollektion (Körper),
POINT2|	2D-Punkt,
TEXT2|	2D-Text,
POLY2|	2D-Polygonzug,
CIRCLE2|	2D-Kreis,
ARC2|	2D-Kreisbogen,
AXIS|	Koordinatenachse und
CURVE	Kurve in Problemkoordianten.

Mit diesen Datentypen können Strukturen von Feldern, aber keine Felder von Strukturen deklariert werden.

< length>

Einige graphische Datentypen erfordern eine Angabe zur Anzahl der aufnehmbaren Elemente:

TEXT, TEXT2 } max. Anzahl aufnehmbarer Zeichen,

POLY, POLY2, CURVE } Anzahl der Zahlentupel,

AXIS max. Anzahl aufnehmbarer Zeichen für Achsenbeschriftung,

SPACE max. Anzahl der volumenbeschreibenden Flächen eines Raumelementes.

< storage class>

An Speicherklassen für die graphischen Daten sind erlaubt:

- AUTOMATIC
- BASED
- CONTROLLED und
- PARAMETER.

Die Speicherklasse PARAMETER muß für alle graphischen Variablen in der Parameterliste angegeben werden, da keine Erkennung der Namen durch den Precompiler möglich ist.

Sind die Parameter dimensioniert, so kann eine Angabe der Grenzen durch '*' nur erfolgen, wenn die Menge der möglichen Indizes jeder Dimension die 1 enthält. Ansonsten muß die untere Feldgrenze angegeben werden.

Die Allokierung von CONTROLLED- und BASED-Variablen erfolgt unter Angabe des vollen Strukturnamens. Die SET-Option kann nicht verwendet werden.

A2.2 Graphische Zuweisung

Die graphische Zuweisung bewirkt die Übertragung von Attributwerten zwischen Objekten gleichen graphischen Types:

< graphical assignment> :: = SET < grevar> = < grex>;

< grex> :: = < grop> |< grevar>

< grop> :: = < grab> |< grat> |< graf-ref> |< grast-ref>

Diese variablen Sprachelemente wurden unter 3.3.2 ebenso erläutert wie die Anwendung der speziellen Zuweisungsform FILL für Polygonzüge und Problemkurven.

A2.3 Graphische Operationen

Eine rein tabellarische Aufstellung der graphischen Operationen, gegliedert nach Builtin-Funktion und Lineartransformationen, wurde bereits in Abschnitt 3.3.3 wiedergegebenen (vgl. auch Abb. 3.8). Die Vielzahl der graphischen Operationen wird im folgenden syntaktisch und semantisch beschrieben.

A2.3.1 Graphische Builtin-Funktionen

< point2>::= POINT2(< lvalue>, < lvalue>)
Diese Builtin-Funktion erzeugt als Ergebnis ihrer Ausführung ein zweidimensionales Punktobjekt, dessen Koordinaten x_1 und x_2 durch die Angaben < lvalue> festgelegt werden. Beispielhafte Aufrufe zeigt Abb. 3.9.

< circle2>::= CIRCLE2(< grex>, < grex>, < grex>, < stringexpr>)|
Zur Erzeugung eines zweidimensionalen Kreisobjektes stehen drei Möglichkeiten bereit. Die obige Form wird durch die Angabe von drei zweidimensionalen Punkten und die Aussage, ob es sich um den In- oder Umkreis zu diesen Punkten handelt, bestimmt. Im Stringausdruck über 'IN' oder 'OUT' erfolgt die Unterscheidung.

CIRCLE2(< grex>, < grex>)|
Diese zweite Form zur Bildung eines zweidimensionalen Kreises ist durch zwei Punkte gegeben, die zum einen den Mittelpunkt zum anderen einen Peripheriepunkt darstellen.

CIRCLE2(< grex>, < lvalue>)
Durch diese Angaben wird ein zweidimensionaler Kreis mittels Zentrum und Radius spezifiziert.

Beispiele zur Verwendung dieser kreisbestimmenden Operationen finden sich in Abb. 3.9.

< arc2> ::= ARC2(< grex>, < avalue>, < avalue>, < lvalue>)|
Ein Kreisbogen wird in dieser Form festgelegt durch die Angaben des Mittelpunktes, der Winkel des Anfangs- und Endpunktes - im mathematisch positiven Sinne gerechnet - und des Radius.

ARC2(< grex>, < grex>, < lvalue>, < stringexpr>)|
Die Verbindung eines Kreisbogens kann auch gegeben sein durch zwei Punkte, den Bogenradius und die Aussage, ob die Verbindung durch den großen ('LARGE') oder kleinen ('SMALL') Bogenschlag erfolgen soll. Die Verbindung geht im mathematisch positiven Sinn vom ersten Punkt aus.

ARC2(< grex>, < grex>, < grex>)|
Die drei Punkte in der Argumentliste dieser Operation beschreiben einen Kreisbogen im positiven Umlaufsinn.

ARC2(< grex>, < grex>, < lvalue>)
In dieser Form wird ein Kreisbogen durch zwei Punkte und die Länge des verbindenden Bogens bestimmt.

Für alle Formen von Builtin-Funktionen zur Erzeugung von Kreisbögen sind Beispiele in Abb. 3.9 aufgeführt.

< text2> ::= TEXT2(< stringexpr>, < grex>, < avalue>)

Ein Textobjekt wird bestimmt durch die Zeichenkette, die es aufnehmen soll und die durch einen beliebigen Stringausdruck spezifiziert wird, die Lage des linken unteren Eckpunktes des ersten Zeichens, dessen Angabe über ein Punktobjekt erfolgt, und die Richtung, unter der ausgehend von diesem Anfang der Text gezeichnet wird. In Abb. 3.12 enthalten die letzten Statements ein Beispiel für diese Operation.

< axis> ::= AXIS(< grex>, < elementexpr>, < elementexpr>, < lvalue>, < avalue>, < stringexpr>, < stringexpr>)

Durch diese Builtin-Funktion wird eine bei Punkt (< grex>) beginnende und unter Winkel (< avalue>) verlaufende Koordinatenachse festgelegt. Die aufnehmbaren Problemwerte bewegen sich zwischen den Angaben des Minimums und Maximums in den beiden Elementausdrücken. Die Länge der Achse wird durch < lvalue> und ihre Beschriftung über das erste < stringexpr> Argumen bestimmt. Mit dem letzten Ausdruck wird entweder eine lineare ('LINEAR') oder logarithmische ('LOGARITHMIC') Skalenteilung ausgewählt.

< xaxis> :: = XAXIS(< grex>, < stringexpr>, < stringexpr>)

Im Gegensatz zur obigen Operation < axis>, bei der alle Angaben zur Koordinatenachse vom Anwender gemacht werden müssen, ist bei dieser Funktion eine automatische Erzeugung einer passenden Achse zu einem Kurvenobjekt (< grex>) oder einer Kollektion davon möglich. Der Beginn (Origin) der x-Achse wird jeweils 3 cm vom linken bzw. unteren Rand der Abbildung angenommen. Die Länge richtet sich nach den Abmessungen des aktuellen Bildfensters. Die beiden Zeichenkettenausdrücke haben dieselbe Bedeutung wie bei < axis>.

< yaxis> ::= YAXIS(< grex>, < stringexpr>, < stringexpr>)
Diese Builtin-Funktion erzeugt automatisch eine Y-Achse. Sie ist ansonsten bedeutungsgleich mit der obigen Operation zur Bestimmung einer X-Achse.

Für alle Möglichkeiten zur Ermittlung von Achsenobjekten gibt Abb. 3.12 einen Überblick.

Während die bisherigen Operationen der Erzeugung zweidimensionaler graphischer Objekte dienten, werden bei den folgenden in Abhängigkeit vom Argumententyp entweder zwei- oder dreidimensional graphische Daten kreiert.

< line> ::= LINE(< grex>, < grex>)
Durch die Angabe zweier Punkte wird die sie verbindende Linie spezifiziert. Handelt es sich bei den Punkten um 2D-Objekte, so ergibt sich eine Verbindung im Zweidimensionalen, andernfalls ist die räumliche Verbindung bestimmt.

< polygon> ::= POLYGON(< grex>)
Diese Operation dient der Erzeugung eines zwei- oder dreidimensionalen Polygonzuges in Abhängigkeit eines Argumentes vom Typ COLLECTION. Das Kollektionsobjekt kann dabei beliebige Kombinationen von Punkten und Polygonzügen in zwei oder drei Dimensionen enthalten. Treten ausschließlich zweidimensionale Objekte in der Kollektion auf, so wird ein 2D-Polygonzug gebildet. Gleiches gilt für 3D-Objekte. Bei gemischter Angabe wird ein zweidimensionales Objekt kreiert, bei dem die dreidimensionalen Anteile unter Beachtung der aktuellen Projektionsbedingungen in ihre zweidimensionale Repräsentation abgebildet werden. Beispiele zur Vereinbarung von Polygonzügen sind in Abb. 3.9 zusammengestellt.

< npoint> ::= NPOINT(< elementexpr>, < grex>)
Durch diese Funktion wird der n-te Punkt, dessen Position durch < elementexpr> spezifiziert wird, aus einem Polygonzug (< grex>) gewonnen. Die Dimensionalität entspricht der des angegebenen Polygonzuges.

Ist die gewünschte Position größer als die Anzahl der Punkte im Objekt, so wir der letzte extrahiert.

< npolygon> ::= NPOLYGON(< elementexpr>, < elementexpr>, < grex>)

Mittels NPOLYGON wird aus einem Polygonzug (< grex>) ein Teilbereich herausgelöst, dessen Beginn und Ende durch die Elementausdrücke festgelegt ist. Die Dimensionalität des Ergebnisses entspricht der des Argumentpolygonzuges.

< refpoint> ::= REFPOINT(< grex>)

Diese Funktion erlaubt die Extraktion eines sogenannten Referenzpunktes aus jedem graphischen Objekt. Die Dimensionalität wird durch das Ausgangsobjekt bestimmt. Dreidimensionale Objekte werden geliefert durch:

- den 1. Punkt eines 3D-Polygonzuges,
- den Anfangspunkt eines 3D-Textes,
- den Mittelpunkt von Kreis und Kugel,
- den Punkt in der Ebene,
- den Achsenpunkt des Zylinders und
- die Kegelspitze.

Zweidimensionale Punkte sind das Ergebnis für

- den 1. Punkt eines 2D-Polygonzuges oder einer Kurve,
- den Anfangspunkt von Texten und Achsen und
- den Mittelpunkt von 2D-Kreisen und -Bögen.

< intersect> ::= INTERSECTION(< grex>, < grex>)

Die Operation errechnet die Schnittpunkte, die beim Durchstoßen mit Polygonzügen oder 2D-Kreisen entstehen. Die Zusammenhänge zwischen Argumenten und Ergbnisobjekten macht folgende Matrix deutlich.

		3D	2D	
	2. Argument / 1. Argument	Polygon	Polygon	Circle
2D	Polygon		Point	Polygon
2D	Circle		Polygon	Polygon
3D	Plane	Point		
3D	Ball	Polygon		
3D	Cylinder	Polygon		
3D	Space	Polygon		

Die Dimension des Ergebnisses entspricht der des 2. Argumentes. Sind zwei Schnittpunkte möglich, so sind diese in einem Polygonzug abgelegt.

Die Ergebnisse der folgenden graphischen Builtin-Funktion sind dreidimensionale Objekte.

< point> ::= POINT(< lvalue>, < lvalue>, < lvalue>)
Die POINT-Funktion erzeugt als Ergebnis ihrer Ausführung ein dreidimensionales Punktobjekt, dessen Koordinaten x_1, x_2 und x_3 durch die Angaben < lvalue> bestimmt sind. Exemplarische Aufrufe enthält Abb. 3.9.

< circle> ::= CIRCLE(< grex>, < grex>, < lvalue>)|
Die Spezifikation eines dreidimensionalen Kreises kann auf drei Wegen vorgenommen werden. Die obigen Angaben beschreiben ihn durch zwei Punkte auf der Achse, von denen der erste als Mittelpunkt interpretiert wird, und den Radius.

CIRCLE(< grex>, < arrayexpr>, < lvalue>)|
In dieser Form wird der Kreis durch den Mittelpunkt

(< grex>), die Angabe der Achsenrichtung als Feld mit Vektorkomponenten und den Radius festgelegt.

CIRCLE(< grex>, < lvalue>)
Die Angabe der Achsenrichtung erfolgt hier über die ersten beiden Punkte eines dreidimensionalen Polygonzuges. Der erste gibt den Mittelpunkt an. Der Radius wird durch < lvalue> bestimmt.

< text> ::= TEXT(< stringexpr>, < grex>, < grex>)
Diese Funktion beschreibt einen Text, dessen Lage und Ausrichtung sich aus der Projektion der dreidimensionalen Punktobjekte ergibt. Die Abbildung des ersten bestimmt den Anfangspunkt und die Verbindung zwischen beiden seine Richtung. Die Zeichenkette selbst ist zweidimensional und wird durch einen beliebigen Stringausdruck spezifiziert.

< plane> ::= PLANE(< grex>, < grex>)|
Eine Ebene ist mathematisch festgelegt durch einen Punkt in der Fläche und die Normalenrichtung. Die verschiedenartigen Beschreibungsmöglichkeiten führen immer auf diese interne Darstellung. In der obigen Form ist das erste Argument der Punkt in der Ebene und das zweite ein Punkt auf der Normalen.

PLANE(< grex>, < grex>, < grex>)|
Die Angabe einer Ebene erfolgt hier durch drei in der Ebene liegende Punkte. Der Umlaufsinn legt die Richtung der Normalen fest (Korkenzieherregel).

PLANE(< grex>, < arrayexpr>)|
Die Ebene wird bestimmt durch einen Punkt (< grex>) und die Normalenrichtung als Feld der Vektorkomponenten < arrayexpr>.

PLANE(< grex>)
Diese Spezifikation benutzt die ersten beiden Punkte eines Polygonobjektes zur Vereinbarung eines Punktes der Ebene und der Richtung der Normalen.

< ball> ::= BALL(< grex>, < lvalue>)
Eine Kugel ist charakterisiert durch Mittelpunkt und Radius. Die Argumente zu obiger Funktion übermitteln diese Größen.

< cylinder> ::= CYLINDER(< grex>, < grex>, < lvalue>)|
Ein Zylinder wird intern durch einen Punkt auf der Achse, die Richtung der Achse und den Radius beschrieben. Diese Angaben werden in ähnlicher Form wie beim Kreis spezifiziert.
Die obige Funktion ermittelt diese geometrischen Daten aus zwei Punkten auf der Mittellinie und dem Radius.

CYLINDER(< grex>, < arrayexpr>, < lvalue>)|
Die Argumente sind ein Punkt auf der Achse (< grex>), ein Feld mit den Komponenten des Richtungsvektors (< arrayexpr>) und der Radius (< lvalue>).

CYLINDER(< grex>, < lvalue>)
Die Werte für die Lage und die Richtung des Zylinders werden hier durch die ersten beiden Punkte eines Polygonzuges festgelegt. Das zweite Argument steht für den Radius.

< cone> ::= CONE(< grex>, < grex>, < lvalue>)|
Diese Funktion dient der Erzeugung eines Kegels aus der Angabe der Spitze und eines Punktes auf der Mittellinie, für den der folgende Radiuswert gilt.

CONE(< grex>, < arrayexpr>, < avalue>)|
Ein Kegel wird hier beschrieben durch die Spitze, die Richtungskomponenten der Rotationsachse und den halben Öffnungswinkel δ.

CONE(< grex>, < avalue>)
Die Lage und die Richtung des Kegels werden durch die ersten beiden Punkte eines Polygonzuges festgelegt. Das zweite Argument spezifiziert den halben Öffnungswinkel δ.

< space> ::= SPACE([±] < grex> $[\pm$ < grex>$]_0^{61}$)
Die Funktion realisiert die in Abschnitt 4.4 beschriebenen Operationen zur Bildung von Raumelementen. Die graphischen Ausdrücke in der Argumentenliste können Flächen oder wiederum Raumelemente erzeugen.
Der '+'-Operator dient der Schnittmengenbildung zwischen den an der Definition beteiligten Flächen bei Normalorientierung.
Der '-'-Operator führt zur Inversion des Definitionbereiches der betroffenen Fläche. Bezogen auf ein Raumelement wirkt nur diese Umkehrung. Beim Aufbau von Körpern kennzeichnet der '-'-Operator in einem der zu vereinigenden Volumina die Berührungsfläche als virtuelle Fläche (vgl. Abschnitt 4.5).
Die Inversion kann nur auf Elementvariable Anwendung finden, da bei den Algorithmen zur Körperanalyse diese Identifikation ausgenutzt wird.

< collection> ::= COLLECTION(< grex> $[+$ < grex>$]_0^{61}$)
Diese Operation findet in zwei unterschiedlichen Versionen Anwendung:

Körperdefinition:
Zur Spezifikation von Körpern wird die Vereinigungsmenge von Raumelementen gebildet. Die Verknüpfung

der Objekte wird in der Argumentliste durch den '+'-Operator gekennzeichnet. Neben Raumelementen können auch bereits definierte Körper (Kollektionen) als Argument angegeben werden. Das Auftreten von virtuellen Flächen wird durch unterschiedliche Vorzeichen in den beteiligten Raumelementen angezeigt.

Zusammenfassung von graphischen Objekten:
Durch die obige Funktion lassen sich beliebige graphische Objekte, die keine Flächen oder Raumelemente sind, in einer Liste auffädeln und so später über einen Namen referieren. Die Liste enthält dabei eine temporäre Kopie der Objekte mit gleichen graphischen Attributwerten.

< niveau> ::= NIVEAU(< arrayexpr>, < arrayexpr>, < arrayexpr>, < elementexpr>)

Diese Funktion erlaubt die Bestimmung von Höhenlinien eines Wertefeldes ($x_1(*,*)$, $x_2(*,*)$, $x_3(*,*)$). Die drei Arrays von Koordinatenwerten sind in den obigen < arrayexpr> repräsentiert. Das letzte Argument gibt den x_3-Wert für die Lage der Höhenebene an. Das Ergebnis der Höhenlinienberechnung können ein oder mehrere Polygonzüge und/oder ein oder mehrere Punkte sein. Alle Objekte werden in einer Kollektion gesammelt und in dieser Form zurückgeliefert. Die Abbildung dieser räumlichen Objekte erfolgt nach den üblichen Projektionsregeln.

A2.3.2 Graphische Transformationen

Eine graphische Transformation verändert aufgrund der in Anhang C wiedergegebenen Beziehungen die geometrischen Attributwerte der graphischen Objekte.

< shift2> ::= SHIFT2(< grex> , < lvalue> , < lvalue>)

Diese Transformation bewirkt die Verschiebung eines Objektes parallel zur x_1- und/oder x_2-Achse.

< scale2> ::= SCALE2(< grex> , < elementexpr> , < elementexpr> [, < grevar>])

Durch die Angabe zweier Faktoren wird die Skalierung eines graphischen Objektes bezüglich des Ursprungs oder eines Punktobjektes (< grevar>) erreicht. Negative Skalierungswerte führen zur Spiegelung an den Koordinatenachsen.

< rotate2> ::= ROTATE2(< grex> , < avalue> [, < grevar>])

Diese Funktion bedingt die Verdrehung eines Objektes im mathematisch positiven Sinn um den Winkel < avalue> um den Koordinatenursprung oder ein Punktobjekt < grevar>.

< shift> ::= SHIFT(< grex> , < lvalue> , < lvalue> , < lvalue>)

Die Verschiebungsoperation erfordert bei einem räumlichen Objekt die Angaben von drei Vektorkomponenten entlang der Koordinatenachsen.

< scale> ::= SCALE(< grex> , < elementexpr> , < elementexpr> ,
< elementexpr> [, < grevar>])

Die Angabe von Skalierungsfaktoren bewirkt die Veränderung der Attributwerte bezogen auf den Ursprung oder einen Punkt < grevar>.

< rotate> ::= ROTATE(< grex> , < avalue> , < avalue> , < avalue> [, < grevar>])

Die drei Winkelangaben bestimmen die Verdrehung eines Objektes um die x_3-, x_2- bzw. x_1-Achse bezogen auf den Ursprung oder einen Punkt < grevar>.

< move> ::= MOVE(< grex>, < grex>)

Diese Funktion dient der Spezifikation einer Verschiebeoperation. Die Komponenten des Verschiebungsvektors werden einem Punktobjekt entnommen, das als zweites Argument angegeben wird. Die Dimensionen von zu verschiebendem Objekt und vom Punkt müssen übereinstimmen.

< moveto> ::= MOVETO(< grex> < grex>)

Während in der vorgenannten Operation ein Punkt den Vektor angab um den das Objekt verschoben werden sollte, so wird festgelegt, wohin die Verschiebung erfolgt.

< switch> ::= SWITCH(< grex>)

Dieser Aufruf ist nur für Flächen anwendbar. Er bewirkt eine permanente Umkehrung der Raumdefinitionen. Bei Ebenen, die zuvor mit Standardorientierung verwendet wurden, liegt z.B. dann der 'materialerfüllte' Bereich auf der Gegenseite der Normalen.

< reduce> ::= REDUCE(< grex>)

Durch REDUCE wird eine Abbildung der 3D-Objekte POINT, TEXT, POLYGON und CIRCLE unter Heranziehung der aktuellen Projektionsmatrix in ihre 2D-Repräsentanten bewirkt. Kreise werden dabei in Polygonzüge gewandelt, um auch Ellipsen, die sich bei nicht senkrechter Anordnung zwischen Kreisachse und Bildebene ergeben, zeichnen zu können.

A2.4 Graphische Ausgabe und deren Kontrolle

A2.4.1 Druckausgabe des Systemzustandes

Durch das folgende Statement wird die Ausgabe des aktuellen Systemzustandes bewirkt:

STATUS [ALL | UNITS | COORDINATEBASE | PROJECTION | PLOT] ;

Die über den Drucker ausgegebene Liste umfaßt Angaben:

- zu Standardlängen- und -winkelangaben (UNITS),
- zur Koordinatenbasis (COORDIN.),
- zu Abbildungsparametern (PROJECTION) und
- zu Inkrementen bei der Kurvenberechnung, zu den Linientypen, zu dem Koordinatenbereich den die graphischen Objekte beanspruchen sowie zu der verfügbaren Papierbreite (PLOT).

Durch eines der obigen Schlüsselwörter kann die Ausgabe auf eine spezielle Charakteristik beschränkt werden. Eine beispielhafte vollständige Systeminformationsliste ist in Abb. 3.10 wiedergegeben.

A2.4.2 Druckausgabe der graphischen Daten

Zu Testzwecken oder zur Protokollierung der graphischen Aufgabe ist häufig eine Druckausgabe der graphischen Daten erforderlich. Das Statement

PRINT(< grex> [, < grex>]*);

druckt in strukturierter Form die Attributwerte der in der Liste aufgeführten Objekte, die durch ihren Namen und - wenn erforderlich - einen Indexwert gekennzeichnet werden. Abb. 3.11 zeigt Beispiele für die Druckausgabe graphischer Objekte.

A2.4.3 Kontrolle des Systemzustandes

Der Veränderung der Systemzustände dient die folgende Anweisung:

CHANGE < what> [, < what>]*;

Die Charakteristiken für

- die Einheiten,
- die Koordinatenbasis,
- die Projektion,
- die Kurvenberechnung und
- die Linientypen

können mit großer Flexibilität geändert werden.

< what> ::= UNITS [LENGTH] [MM | CM | M | INCH | FOOT | YARD] |
[ANGLE] [DEGREE | RADIAN]

Diese Angaben legen die Standardeinheiten für Längen- bzw. Winkelangaben fest. Nach der Systeminitialisierung sind MM und DEGREE aktiv. Durch die aktuelle Einheitenkennzeichnung in < lunit> oder < aunit> werden die Standardwerte überschrieben.

COORDINATEBASE < lvalue>, < lvalue>, < lvalue> |
Durch diesen Befehl wird für alle folgenden Koordinatenangaben ein neuer Ursprung definiert.

PROJECTION [PARALLEL | CENTRAL | PI_NORMAL | PI_POINT] < lvalue>, < lvalue>, < lvalue> |

Diese Anweisung erlaubt die Festlegung der zur Projektion von räumlichen Koordinaten erforderlichen Angaben. Als Standardansicht ist vom System eine senkrechte Parallelprojektion in die im Ursprung befindliche (x_2,x_3)-Ebene vorgesehen. Bei Parallelprojektion werden die Richtungskomponenten des Betrachtungsvektors angegeben. Eine Normierung auf den Einheitsvektor erfolgt intern. Für die Zentralprojektion bestimmen die Koordinatenangaben < lvalue> das Projektionszentrum. Bei beiden Projektionsarten ist die Bildebene π noch frei wählbar. Sie wird durch einen Punkt (PI_POINT) und die Normalenrichtung (PI_NORMAL) spezifiziert. Es ist darauf zu achten, daß ein reales Bild entstehen kann, d.h. Projektionsstrahlen dürfen nicht parallel zur Bildebene verlaufen.

INCREMENT [LINEAR < lvalue>] |
[ANGLE < avalue>]

Die Effektivität der Kurvenberechnung wird stark von den Inkrementen beeinflußt. Der Anwender kann durch die obige Anweisung eine objektbezogene Optimierung vornehmen. Vom System werden zunächst Schritte von 8 mm bzw. 12 Grad angenommen.

LINETYPE [VISIBLE] [THROUGH]
[INVISIBLE] [DASHED]
[CENTER] [CENTER]
[OMITTED]

Durch die Kombination dieser Darstellungsmerkmale läßt sich die Ausgabe von Linien modifizieren. Die Darstellung unsichtbarer Kanten (INVISIBLE) erfolgt durch die Standardfestlegung in gestrichelter Form (DASHED). Bei Angabe von OMITTED werden sie weggelassen. Wird THROUGH spezifiziert, so erfolgt keine Überprüfung auf Unsichtbarkeit und alle Körperkanten erscheinen durchgezogen.

Für eine sinnvolle Ausnutzung der Zeichenfläche wird vom System die Anordnung der einzelnen Abbildungen überwacht. Die Größe dieser Bildfenster wird vom Anwender festgelegt und alle nachfolgenden Zeichenbefehle werden auf das Verlassen dieses Bereiches überprüft. Der Standardwert für das verfügbare Zeichenfeld ist DIN A4 hoch. Der Anwender spezifiziert und eröffnet ein neues Bildfenster durch:

OPEN PLOT [DIN A < elementexpr> [BROAD] | [WIDTH] < lvalue> X [HEIGHT] < lvalue>] ;

Die Abmessungen des Bildes werden dabei durch ein DIN-Format oder freie Breiten- und Höhenangaben festgelegt.

A2.4.4 Kontrolle des Objektzustandes

Die Angaben der graphischen Objekte erfolgt unter Beachtung der Darstellungsattribute. Diese objektspezifischen Angaben sind vom Anwender durch das folgende Statement spezifizierbar:

EDIT [ATTRIBUTES] < attr> [, < attr>]* FOR(< grevar> [, < grevar>]*);

Die Möglichkeit zur Veränderung der Darstellungsattribute betreffen:

< attr> ::= SYMBOL < elementexpr> |
Durch diese Angabe wird bei Punkten, Polygonzügen oder Kurven das zur Kennzeichnung verwendete Symbol ausgewählt. Der Wert des Elementausdruckes korrespondiert mit den Symbolnummern einer gerätespezifischen Plottersoftware.

HEIGHT < lvalue> |
Die Höhe von Punktsymbolen und Texten wird in dieser Form vereinbart.

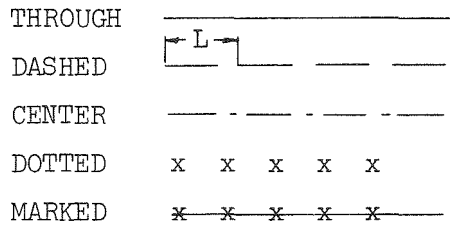
EVERY < elementexpr> |
Dieser Ausdruck gibt jeden n-ten Punkt eines Polygonzuges oder einer Kurve an, der durch ein Punktesymbol gekennzeichnet werden soll.

```
[OPEN  ]  |
[CLOSED]
```

Soll bei Polygonzügen ein Schießen erfolgen, d.h.der letzte Punkt mit dem ersten wieder verbunden werden, so wird CLOSED angegeben.

```
LINETYPE  [THROUGH]  |
          [DASHED ]
          [CENTER ]
          [DOTTED ]
          [MARKED ]
          [OMITTED]
```

Die Art der Liniendarstellung wird durch die obigen Angaben ausgewählt. Die Normalausgabe von Polygonzügen und Kurven erfolgt in durchgezogener Form. Zur Unterscheidung von Linien - vor allem in Diagrammen - ist folgende Kennzeichnung möglich:



Die Länge L für die Segmente einer gestrichelten oder strichpunktierten Linie wird durch

```
LENGTH <lvalue> |
```

vereinbart.

Die Darstellung der gepunkteten und markierten Linien benutzt die Angaben zu SYMBOL, HEIGHT und EVERY zur Ausgabe entsprechender Punktesymbole.

Die folgenden Objektattribute beziehen sich auf Koordinatenachsen.

```
[MINIMUM] <elementexpr>|
[MAXIMUM]
```

Diese Attributspezifikation legt den Problemwertebereich einer Koordinatenachse fest, in den die darzustellenden Werte fallen müssen.

[NORMAL
LOGARITHMIC] |

Hierdurch wird die Art der Koordinatenachse definiert.

[RIGHT
LEFT]

Die Beschriftung erscheint im mathematischen Umlaufsinn vor (RIGHT) bzw. hinter (LEFT) der Koordinatenachse.

A2.4.5 Graphische Ausgabe

Die graphische Ausgabe von Objekten wird durch das PLOT-Statement veranlaßt und erfolgt unter Interpretation der aktuellen System- und Objektzustände in das zuletzt eröffnete Bildfenster. Diese Anweisung wurde bereits in den Abschnitten 3.3.4.3 und 4.7 in ihrer Bedeutung und Anwendung erläutert. An dieser Stelle sollen neben der vollständigen Syntax nur einige ergänzende Angaben zur Anwendung folgen.

Die Varianten des PLOT-Statements lauten:

PLOT (< grex> [, < grex>]*);

Diese Standardform des Plotaufrufes bewirkt die Ausgabe graphischer Objekte, die über graphische Ausdrücke spezifiziert sind.

PLOT DIAGRAMME WITH [OUT] [AXIS] < grevar>
[AND] < grevar> OF (< grevar> [, < grevar>]*);

Diese Anweisung ermöglicht die Ausgabe von Kurvenobjekten in Problemkoordinaten. Eine beispielhafte Anwendung dieser Fähigkeit zeigt Abb. 3.12.

PLOT RELIEF [OF] < ident> [(< index> [, < index>]*) [.]]*
[GENERATED [BY] < arrayexpr> [ALONG] < stringexpr>]
[SCALED [BY] < arrayexpr>]
[LINES < stringexpr>];

Dieser PLOT-Befehl dient der perspektivischen Darstellung von Problemwertefeldern unter Anwendung eines einfachen Sichtbarkeitskriteriums. In Abb. 3.13 sind die Möglichkeiten dieser Darstellungsformen verdeutlicht. Das darzustellende Feld muß mindestens dreifach dimensioniert sein. Die Festlegung der Indizes für den zu betrachtenden Ausschnitt erfolgt durch:

< index> :: = * | ** | < elementexpr>.

Da es sich bei den Werten des Feldes um Problemwerte und nicht um geometrische Größen handelt, sind für deren Umrechnung zusätzliche Optionen vorgesehen.

Die Umrechnung erfolgt durch die Angabe eines Skalierungsfeldes (< arrayexpr>), dessen Elemente mit den Komponenten des darzustellenden Feldes (< ident>) gemäß:

```
IDENT(,,,I)=IDENT(,,,I)*ARRAY_S(I);
```

multipliziert werden.

Sollen die Werte des Feldes entlang einer Koordinate, wobei die Angabe des entsprechenden Indexes in einem String erfolgt, mit einem festen Inkrement generiert werden, so muß dies z.B. über folgende Angaben spezifiziert werden:

```
GENERATED BY ARRAY_G ALONG '2'
SCALED BY ARRAY_S.
```

Diese Angaben führen intern zur Berechnung von Problemwerten nach

```
X(,,I,2)=ARRAY_G(2)+(I-1)*ARRAY_S(2)
```

für alle Elemente der durch 'I' gekennzeichneten Dimension.

Durch die Angabe von

```
LINES 'ABOVE'
```

wird die Ausgabe auf die bei Ansicht von oben erkennbaren Rasterlinien beschränkt.

PLOT

SHADE [OF] < grevar> [WITH < grex>]
[IN] SURFACES (< grevar>[,< grevar>]*);

Mit dieser Variante des PLOT-Befehls wird die Schraffur körperbegrenzender Flächen bewirkt. Das räumliche Objekt wird durch eine graphische Elementvariable gekennzeichnet. Ist die zu schraffierende Fläche eine Ebene, so werden alle erforderlichen Angaben automatisch generiert. Für alle anderen Flächentypen wird die Schraffur durch eine Ebene festgelegt, die in

der angegebenen Fläche einen Schnitt in der gewünschten Richtung der Schraffurlinien erzeugt.

Eine exemplarische Ausgabe dieser Anwendung ist in Abb. 4.21 wiedergegeben.

B. Nomenklatur

Matrizen	sind zweimal unterstrichen $\underline{\underline{\ }}$
Vektoren	sind einmal unterstrichen $\underline{\ }$
$\underline{e}, \underline{a}, \underline{b}, \underline{n}, \underline{h}$	Einheitsvektoren
$\underline{x}, \underline{m}, \underline{p}, \underline{z}$	Radiusvektoren
$\underline{e}_i$	orthonormierte Basisvektoren des dreidimensionalen euklidischen Raumes E^3 (Rechtssystem)
$\underline{x}_i = x_i \underline{e}_i$	Komponenten eines Radiusvektors
$\underline{x} = \sum \underline{x}_i$	abgekürzte Schreibweise für $\underline{x} = \underline{x}_1 + \underline{x}_2 + \underline{x}_3$
$\underline{e}'_i$	orthonormierte Basisvektoren in der Bildebene
$\underline{x}'_i$	Bildebenenkoordinaten des virtuellen Bildes
$\underline{e}''_i$	orthonormierte Basisvektoren in der Zeichenebene Z
$\underline{x}''_i$	Zeichenebenenkoordinaten des reellen Bildes
α, β, γ	Drehungswinkel um die Koordinatenachsen $\underline{e}_3$, $\underline{e}_2$ und $\underline{e}_1$ im mathematisch positiven Sinne
$\underline{V}$	Verschiebungsvektor }
$\underline{S}$	Skalierungsvektor } bei Lineartransformation
$\underline{\underline{R}}$	Rotationsmatrix }
T	transponiert
$\top$	Bildebene
Z	Zeichenebene
$\perp$	senkrecht
$\parallel$	parallel
$\uparrow\downarrow$	antiparallel

Indizes

i	Komponentenkennzeichnung bei Vektoren
E	Ebene
Ku	Kugel
Z	Zylinder

Ke	Kegel
K	Kreis
G	Gerade
V, S, R	lineartransformierte Koordinaten
H	homogen

Parameter

u, v, w, t	Parameterangaben bei der Durchdringungs- oder Umrißbestimmung

spezielle Vektoren

$\underline{z}$	Projektionsrichtung bei Parallel- bzw. Projektionszentrum bei Zentralprojektion.
$\underline{p}_E$	Punkt der Ebene
$\underline{n}_E$	Normale der Ebene
$\underline{m}_{Ku}$	Mittelpunkt der Kugel
$\underline{m}_Z$	Achsenpunkt eines Zylinders
$\underline{n}_Z$	Richtung der Zylinderachse
$\underline{p}_{Ke}$	Kegelspitze
$\underline{m}_{Ke}$	Punkt der Kegelachse, für den die Radiusangabe gilt

spezielle Skalare

r_{Ku}	Radius der Kugel
r_Z	Radius des Zylinders
r_{Ke}	Radius des Kegels
δ	halber Öffnungswinkel des Kegels

Rechenvorschriften

$\underline{c} = \underline{a} \times \underline{b}$	Vektorprodukt
$d = \underline{a}\ \underline{b}$	Skalarprodukt

C. Lineartransformation

C1. Räumliche Lineartransformation von Objekten

In fast allen graphischen Systemen werden die Lineartransformationen für Verschiebungen, Skalierungen und Verdrehungen in einer Matrix zusammengefaßt, die zudem auch die Koeffizienten für eine perspektivische Abbildung - meist nur Normalprojektion - enthält. Bei dieser Methode wird also die Transformation der Objektkoordinaten erst bei Ausführung der Abbildung vollzogen. In GIPSY sind Lineartransformationen unabhängig von einer späteren Projektion. Um eine effektivere Anwendung zu erreichen, werden die Transformationen nicht in einer Matrix angewendet, sondern z.B. für Verschiebung und Skalierung als Vektoraddition bzw. -multiplikation realisiert.
Die Verschiebung ('shift') ergibt sich dann aus

$$\underline{x}_V = \underline{x} + \underline{V} ,$$

wobei $\underline{V}$ der Verschiebungsvektor

$$\underline{V} = (V_1, V_2, V_3)$$

ist.

Zur Skalierung ('scale') ist lediglich eine komponentenweise Multiplikation

$$\underline{x}_S = \underline{x} \cdot \underline{S}$$

mit

$$\underline{S} = (S_1, S_2, S_3)$$

erforderlich.

Für die Rotation ('rotation') der Raumpunkte $\underline{x}$ im mathematisch positiven Sinne um die Winkel α,β und γ - bezogen auf die Basisvektoren $\underline{e}_3$, $\underline{e}_2$ und $\underline{e}_1$ - wird die Matrix

$$\underline{\underline{R}} = \left(\begin{array}{c|c|c} \cos\alpha\cos\beta & \sin\alpha\cos\beta & -\sin\beta \\ \hline \cos\alpha\sin\beta\sin\gamma & \sin\alpha\sin\beta\sin\gamma & \\ -\sin\alpha\cos\gamma & +\cos\alpha\cos\gamma & \cos\beta\sin\gamma \\ \hline \cos\alpha\sin\beta\cos\gamma & \sin\alpha\sin\beta\cos\gamma & \\ +\sin\alpha\sin\gamma & -\cos\alpha\sin\gamma & \cos\beta\cos\gamma \end{array}\right)$$

zur Ermittlung von

$$\underline{x}_R = \underline{x} \quad \underline{\underline{R}}$$

herangezogen.

Bei der Anwendung dieser Lineartransformationen auf die flächenbestimmenden Attribute ist folgendes zu beachten:

Lineartr. / Aktion auf	Verschiebung	Skalierung	Rotation
alle vektoriellen Größen		X	X
alle nicht normierten vekt. Größen	X		
alle normierten vekt. Größen neu normieren		X	

Abb. C.1: Einfluß der Lineartransformationen auf Flächenattribute

C2. Transformationen in der Bildebene

Von den Benutzern graphischer Systeme wird häufig neben flexiblen Abbildungsmechanismen eine zusätzliche Möglichkeit zur Manipulation des projizierten Bildes gefordert, wenn

- Koordinaten des Bildes außerhalb des gewählten Betrachtungsfensters liegen,
- Objektabmessungen keine Zeichnung mit vernünftigen Maßen erlauben würden und
- Verschiebungen von Teilen des gleichen Objektraumes gegeneinander gewünscht werden (Explosionsdarstellung).

Zu diesem Zwecke kann der Abbildung eines Objektes eine beliebige lineare Transformation im Zweidimensionalen nachgeschaltet werden.
Diese Transformation entspricht der Transformation des virtuellen Bildes in einen reellen Bildraum.

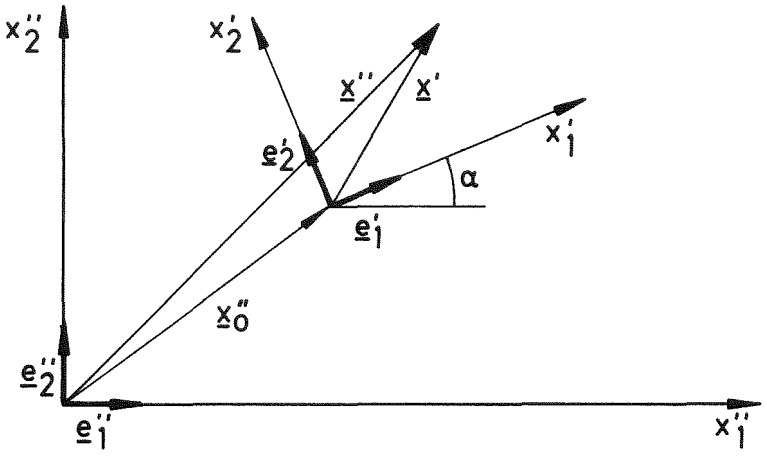


Abb. C.2: Transformation im Bildraum

Für den Punkt $\underline{x}''$ im reelen Bildraum gilt

$$\underline{x}'' = \underline{x}_o'' + \underline{x}'.$$

Unter Anwendung der Komponentenschreibweise wird hieraus

$$\sum_i^2 \underline{e}_i'' \left(x_i'' - x_{oi}'' \right) = \sum_i^2 \underline{e}_i' x_i'$$

Nach Multiplikation mit $\underline{e}_i''$ und Anwendung der folgenden Beziehungen

$$\underline{e}'_i \cdot \underline{e}''_i = \cos\alpha, \quad \underline{e}''_1 \cdot \underline{e}'_2 = -\sin\alpha \text{ und } \underline{e}''_2 \cdot \underline{e}'_1 = \sin\alpha$$

sowie einer zusätzlichen Skalierung $\underline{S}' = (S'_1, S'_2)$ der Koordinatenachsen des virtuellen Bildes ergibt sich für die Koordinaten im reelen Bildraum

$$X_1'' = X_{0_1}'' + S_1' X_1' \cos\alpha - S_2' X_2' \sin\alpha$$

$$X_2'' = X_{0_2}'' + S_2' X_1' \sin\alpha + S_2' X_2' \cos\alpha \,.$$

Diese Gleichungen lassen sich auch in Matrixform für homogene Koordinaten schreiben:

$$\underline{X}_H'' = \underline{X}_H' \cdot \underline{\underline{TM}}$$

wobei

$$\underline{\underline{TM}} = \left(\begin{array}{c|c|c} 1 & X_{0_1}'' & X_{0_2}'' \\ \hline 0 & S_1' \cos\alpha & S_1' \sin\alpha \\ \hline 0 & -S_2' \sin\alpha & S_2' \cos\alpha \end{array}\right)$$

D. Abbildungsgesetz

Dieses Kapitel enthält eine Zusammenfassung der geometrischen Definitionen und Beziehungen, die zur Darstellung von Raumpunkten benötigt werden. Alle geometrischen Operationen sollen (mit einer einzigen Ausnahme) in inhomogenen Koordinaten ausgeführt werden. Die Basis dieses dreidimensionalen euklidischen Raumes ist ein orthonormiertes Rechtssystem.

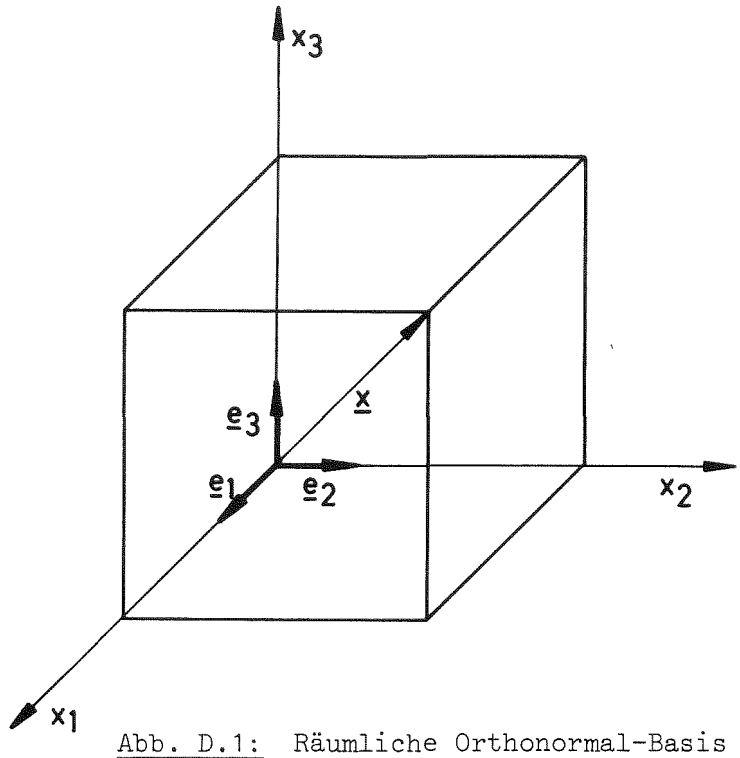


Abb. D.1: Räumliche Orthonormal-Basis

Die Basisvektoren $\underline{e}_1$, $\underline{e}_2$ und $\underline{e}_3$ stehen also paarweise aufeinander senkrecht und haben folgende Komponenten

$$\underline{e}_1 = (1,0,0), \quad \underline{e}_2 = (0,1,0) \text{ und } \underline{e}_3 = (0,0,1).$$

Ein Punkt des Raumes wird durch einen Ortsvektor beschrieben

$$\underline{x} = x_1 \underline{e}_1 + x_2 \underline{e}_2 + x_3 \underline{e}_3 = (x_1, x_2, x_3) \qquad (d1)$$

Die Behandlung dreidimensionaler Objekte erfordert bei ihrer Darstellung die Reduktion der Dimension auf die des Bildraumes. Hier sind nur Projektionen von Interesse, bei denen ein Projektionsraum E^0 (Punkt) mittels eines projizierenden Raumes E^1 (Gerade) in einen Bildraum E^2 (Ebene) abgebildet wird.

Diese Arbeit, die ins Aufgabengebiet der Darstellenden Geometrie fällt, wurde bisher von synthetisch - konstruktiven Methoden bestimmt, da die bei einer rechnerischen Behandlung anfallenden Datenmengen den ohne technische

Hilfsmittel beherrschbaren Umfang sprengt. Erst der Einsatz moderner Rechenanlagen führte zur Anwendung rechnerischer Abbildungsgesetze und damit zu exakten und mit dichten Punktefolgen konstruierten Abbildungen. Für fast alle realisierten graphischen Systeme wurden aber bisher Abbildungsalgorithmen implementiert, die entweder allgemeine Axonometrien oder Normalprojektionen beinhalten. Im folgenden soll die Abbildungsgleichung für beliebige Zentral- bzw. Parallelprojektionen entwickelt werden, die an in /63/ entwickelte Methoden anknüpft.

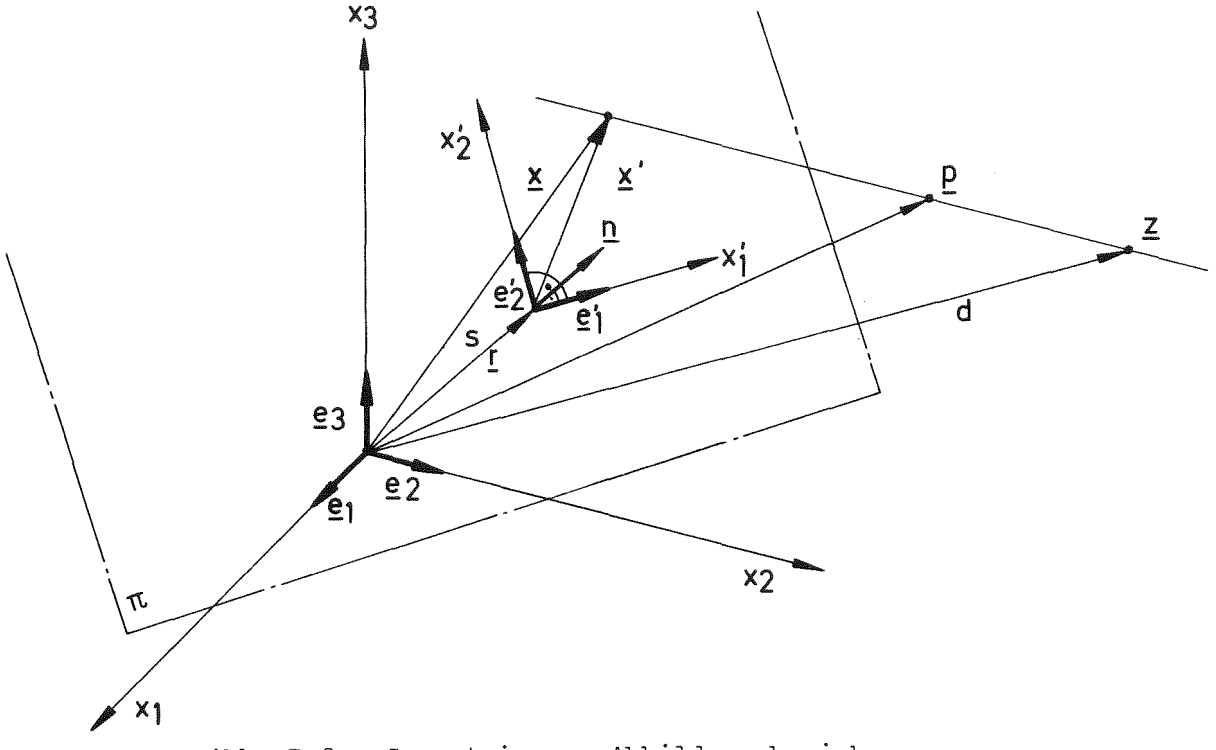


Abb. D.2: Geometrie zur Abbildungsbeziehung

Die Abbildung von Kurven des dreidimensionalen euklidischen Raumes vollzieht sich als eine Projektion von Punkten $\underline{p}$ auf dieser Kurve in eine zweidimensionale euklidische Bildebene π.

Die Bildebene π soll durch einen Punkt $\underline{r}$ gehen, der zum Koordinatenursprung einen Abstand s hat, und die Normalenrichtung $\underline{n}$ besitzen.
Diese Ebene, deren Gleichung sich zu

$$(\underline{x} - \underline{r})\,\underline{n} = 0 \qquad \text{(d2)}$$

ergibt, muß mit dem Projektionsstrahl

$$\underline{x} = \underline{p} + t\,(\underline{z} - \underline{p}) \qquad \text{(d3)}$$

zum Schnitt gebracht werden.

Da $\underline{x}'$ das Bild von $\underline{p}$ in Π ist, ergibt sich durch Einsetzen eine Beziehung für t, die eine Aussage für den Bildpunkt liefert.

$$\underline{X} = \frac{((\underline{z}\,\underline{n}) - (\underline{\tau}\,\underline{n}))\underline{p} + ((\underline{\tau}\,\underline{n}) - (\underline{p}\,\underline{n}))\underline{z}}{(\underline{z} - \underline{p})\,\underline{n}} \tag{d4}$$

Diese Gleichung definiert einen Raumpunkt. Für die zeichnerische Darstellung des Ergebnisses benötigen wir eine Beziehung, die den Punkt $\underline{x}'$ mit den Basisvektoren $\underline{e}'_1$ und $\underline{e}'_2$ des Koordinatensystems in der Bildebene verknüpft. In der Bildebene gilt

$$\underline{x}' = x'_1\,\underline{e}'_1 + x'_2\,\underline{e}'_2 = \underline{x} \tag{d5}$$

In dieser Gleichung sind die Basisvektoren $\underline{e}'_1$ und $\underline{e}'_2$ noch frei wählbar. Einfachste Methode zur Bestimmung von $\underline{e}'_1$ erfolgt aus der Schnittgeraden der Bildebenen Π mit der (x_1, x_2)-Ebene.

Die Ebenengleichungen für

$$\begin{aligned} \Pi: \quad & n_1 x_1 + n_2 x_2 + n_3 x_3 = 0 \quad \text{und} \\ (x_1, x_2): \quad & x_3 = 0 \quad \text{liefern} \end{aligned} \tag{d6}$$

nach Normierung

$$\underline{e}'_1 = \frac{1}{\sqrt{n_1^2 + n_2^2}}\,(-n_2, n_1, 0) \tag{d7}$$

Die zweite Komponente ergibt sich aus der vektoriellen Multiplikation von $\underline{e}'_1$ mit der Normalen $\underline{n}$ der Bildebene

$$\underline{e}'_2 = \underline{n} \times \underline{e}'_1 \tag{d8}$$

zu

$$\underline{e}'_2 = \frac{1}{\sqrt{n_1^2 + n_2^2}}\,(-n_1 n_3, -n_2 n_3, n_1^2 + n_2^2) \tag{d9}$$

Mittels dieser Basisvektoren wird aus (d4) und (d5) nach Multiplikation mit $\underline{e}'_1$

$$\underline{X}'_1 = \frac{(\underline{z}\,\underline{n} - \underline{\tau}\,\underline{n})(\underline{p}\,\underline{e}'_1) + (\underline{\tau}\,\underline{n} - \underline{p}\,\underline{n})(\underline{z}\,\underline{e}'_1)}{(\underline{z} - \underline{p})\,\underline{n}} \tag{d10}$$

Die in dieser Gleichung auftretenden Skalarprodukte berechnen sich nach Einführung der folgenden Abkürzungen

$$|\underline{n}| = \sqrt{n_1^2 + n_2^2 + n_3^2} = 1,$$

$$\lambda_i^2 = 1 - n_i^2,$$

$$\underline{t}\,\underline{n} = d \quad \text{und}$$

$$\underline{r}\,\underline{n} = s$$

zu

$$X_1' = \frac{s(t_2 n_1 - t_1 n_2) + p_1(s n_2 - t_2 \lambda_3^2 - t_3 n_2 n_3) + p_2(-s n_1 + t_1 \lambda_3^2 + t_3 n_1 n_3) + p_3 n_3 (t_1 n_2 - t_2 n_1)}{\lambda_3 d - p_1 n_1 \lambda_3 - p_2 n_2 \lambda_3 - p_3 n_3 \lambda_3} \qquad \text{(d11)}$$

Diese Gleichung für die erste Bildkoordinate läßt sich weiter vereinfachen, wenn man rein formal für $\underline{p}$ und $\underline{z}$ homogene Koordinaten einführt, was zudem den Vorteil hat, daß auch die Parallelprojektion behandelt werden kann.

Für $\underline{p}$ und $\underline{z}$ gilt dann

$$\underline{p}_H = (p_o, p_1, p_2, p_3),$$

$$\underline{z}_H = (z_o, z_1, z_2, z_3)$$

Die erste homogene Komponente von $\underline{p}_H$ und $\underline{z}_H$ ist bei Zentralprojektion gleich 1, während bei der Parallelprojektion z_o den Wert 0 annimmt. Die inhomogenen Koordinaten errechnen sich aus

$$\underline{p} = \left(\frac{p_1}{p_o}, \frac{p_2}{p_o}, \frac{p_3}{p_o}\right) \qquad \text{und} \qquad \underline{t} = \left(\frac{t_1}{t_o}, \frac{t_2}{t_o}, \frac{t_3}{t_o}\right)$$

Die Abbildungsgleichung erhält dann die Form

$$X_1' = \frac{\underline{c}_1 \underline{p}_H}{\underline{c}_o \underline{p}_H} \qquad \text{(d12)}$$

Die Komponenten der Vektoren $\underline{c}_1$ und $\underline{c}_o$ zur Errechnung der ersten Bildkoordinate werden in Abb. D.3 aufgeführt.

Ähnlich wie für die Gleichung (d10) wird eine Beziehung für x'_2 durch Multiplikation von (d5) mit dem Basisvektor $\underline{e}'_2$ gefunden:

c_{ij} (i \ j)	0	1	2	3
0	$\lambda_3 d$	$-n_1\lambda_3 z_o$	$-n_2\lambda_3 z_o$	$-n_3\lambda_3 z_o$
1	$s(z_2n_1-z_1n_2)$	$sn_2z_o-z_3n_2n_3-z_2\lambda_3^2$	$-sn_1z_o+z_3n_1n_3+z_1\lambda_3^2$	$n_3(z_1n_2-z_2n_1)$
2	$-s(z_2n_2n_3+z_1n_1n_3-z_3\lambda_3^2)$	$n_1(sn_3z_o-z_3)$	$n_2(sn_3z_o-z_3)$	$z_1n_1+z_2n_2-s\lambda_3^2z_o$

Abb. D.3: Komponenten der Vektoren $\underline{C}_o$, $\underline{C}_1$ und $\underline{C}_2$ aus Gleichung (d12) und (d15) für $(n_1,n_2)\neq(0,0)$

c_{ij} (i \ j)	DIN 6							
	0	1	2	3	0	1	2	3
0	$-z_3$	0	0	z_o	z_3	0	0	$-z_o$
1	$-z_2s$	0	sz_o-z_3	z_2	z_1s	$-sz_o+z_3$	0	$-z_1$
2	z_1s	$-sz_o+z_3$	0	$-z_1$	z_2s	0	$-sz_o+z_3$	$-z_2$

Abb. D.4: Komponenten der Vektoren $\underline{C}_o$, $\underline{C}_1$ und $\underline{C}_2$ aus Gleichung (d12) und (d15) für $\underline{n} = \underline{e}_3$

$$x'_2 = \frac{(\underline{t}\,\underline{n} - \underline{\tau}\,\underline{n})(\underline{p}\,\underline{e}'_2) + (\underline{\tau}\,\underline{n} - \underline{p}\,\underline{n})(\underline{t}\,\underline{e}'_2)}{(\underline{t} - \underline{p})\,\underline{n}} \tag{d13}$$

Unter Verwendung der obigen Abkürzung folgt

$$x'_2 = \frac{s(t_2 n_2 n_3 - t_1 n_1 n_3 + t_3 \lambda_3^2) + p_1 n_1 (s n_2 - t_3)}{+ p_2 n_2 (s n_3 - t_3) + p_3 (t_1 n_1 + t_2 n_2 - s\lambda^2)} \tag{d14}$$
$$\lambda_3 d - p_1 n_1 \lambda_3 - p_2 n_2 \lambda_3 - p_3 n_3 \lambda_3$$

In Analogie zur obigen Herleitung für x'_1 erhält man

$$x'_2 = \frac{\underline{c}_2\,\underline{p}_H}{\underline{c}_0\,\underline{p}_H} \tag{d15}$$

Die Komponenten der Vektoren $\underline{c}_2$ und $\underline{c}_0$ sind in Abb. D.3 zusammengefaßt. Zur Bestimmung der Basisvektoren in (d6) wurde die Schnittgerade der Bildebene mit der (x_1, x_2)-Ebene herangezogen. Bei der Anwendung des so gefundenen Bildkoordinatensystems muß deshalb sichergestellt sein, daß die Komponenten $\underline{n}_1$ und $\underline{n}_2$ der Bildebenennormalen nicht gleichzeitig 0 werden, da wegen der Parallelität dann keine Schnittgerade entsteht. Zur Beseitigung des Konflikts werden in diesem Sonderfall die Basisvektoren des Bildraumes $\underline{e}'_1$ und $\underline{e}'_2$ fest mit den entsprechenden Komponenten der räumlichen Basis verbunden. Die Bildebenennormale $\underline{n}$ zeigt in die $\underline{e}_3$-Richtung, sodaß sich bei senkrechter Betrachtung ein Grundriß bzw. eine Draufsicht ergibt. Für die Ansicht ist nach DIN 6 eine Anordnung nach festen Regeln vorgeschrieben. Aus Abb. D.5 werden die Annahmen deutlich.

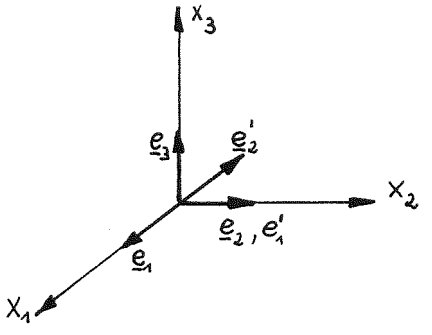


Abb. D.5: Basisvektoren der Bildkoordinaten für Grundriß ($\underline{n} = \underline{e}_3$, DIN 6)

Mit der orthonormierten Basis $\underline{e}'_1 = (0, 1, 0)$ und $\underline{e}'_2 = (-1, 0, 0)$ und den Gleichungen (d10) und (d13) ergeben sich die Bildkoordinaten zu

$$x'_1 = \frac{-z_2 S + p_2 (S - z_3) + p_3 z_2}{-z_3 + p_3} \qquad \text{(d16)}$$

und

$$x'_2 = \frac{z_1 S - p_1 (S - z_3) - p_3 z_1}{-z_3 + p_3} \qquad \text{(d17)}$$

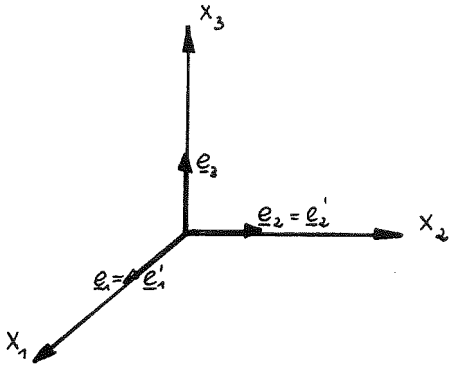


Abb. D.6: Basisvektoren der Bildkoordinaten für Grundriß ($\underline{n} = \underline{e}_3$)

Bei einer Anordnung der Bildebene nach Abb. D.6 sind

$$\underline{e}'_1 = (1,0,0) \text{ und } \underline{e}'_2 = (0,1,0).$$

Bei Anwendung derselben Grundbeziehungen errechnen sich x'_1 und x'_2 aus

$$x'_1 = \frac{z_1 S + p_1 (z_3 - S) - p_3 z_1}{z_3 - p_3} \qquad \text{(d18)}$$

$$x'_2 = \frac{z_2 S + p_2 (z_3 - S) - p_3 z_2}{z_3 - p_3} \qquad \text{(d19)}$$

Nach Umschreiben dieser Beziehungen mit homogenen Koordinaten für $\underline{p}$ und $\underline{z}$ erhält man ebenfalls Gleichungen der Form (d12) und (d15). Die Vektoren $\underline{C}_0$, $\underline{C}_1$ und $\underline{C}_2$ für diesen Sonderfall sind in Abb. D.4 zusammengestellt.

Die Lage von räumlichen Basisvektoren, Bildebene und Projektionsrichtung wird in Abb. D.7 für Parallelprojektion in die Auf-, Seiten- und Grundrißebene verdeutlicht. Die schraffierten Flächen zeigen an, welche Bildpunk-

te ohne weitere Manipulationen in der Bildebene im positiven Bereich liegen. Eine Umkehrung der Projektionsrichtung hat keinen Einfluß auf die Abbildung. Für den Sonderfall $\underline{n} \parallel \underline{e}_3$ besteht keine Abhängigkeit der Basisvektoren des Bildes von $\underline{n}$.

Beide Abbildungsverfahren lassen sich auch durch eine Rotation im Bildraum ineinander überführen. Die hierzu erforderlichen Gleichungen werden im Anhang C angegeben.

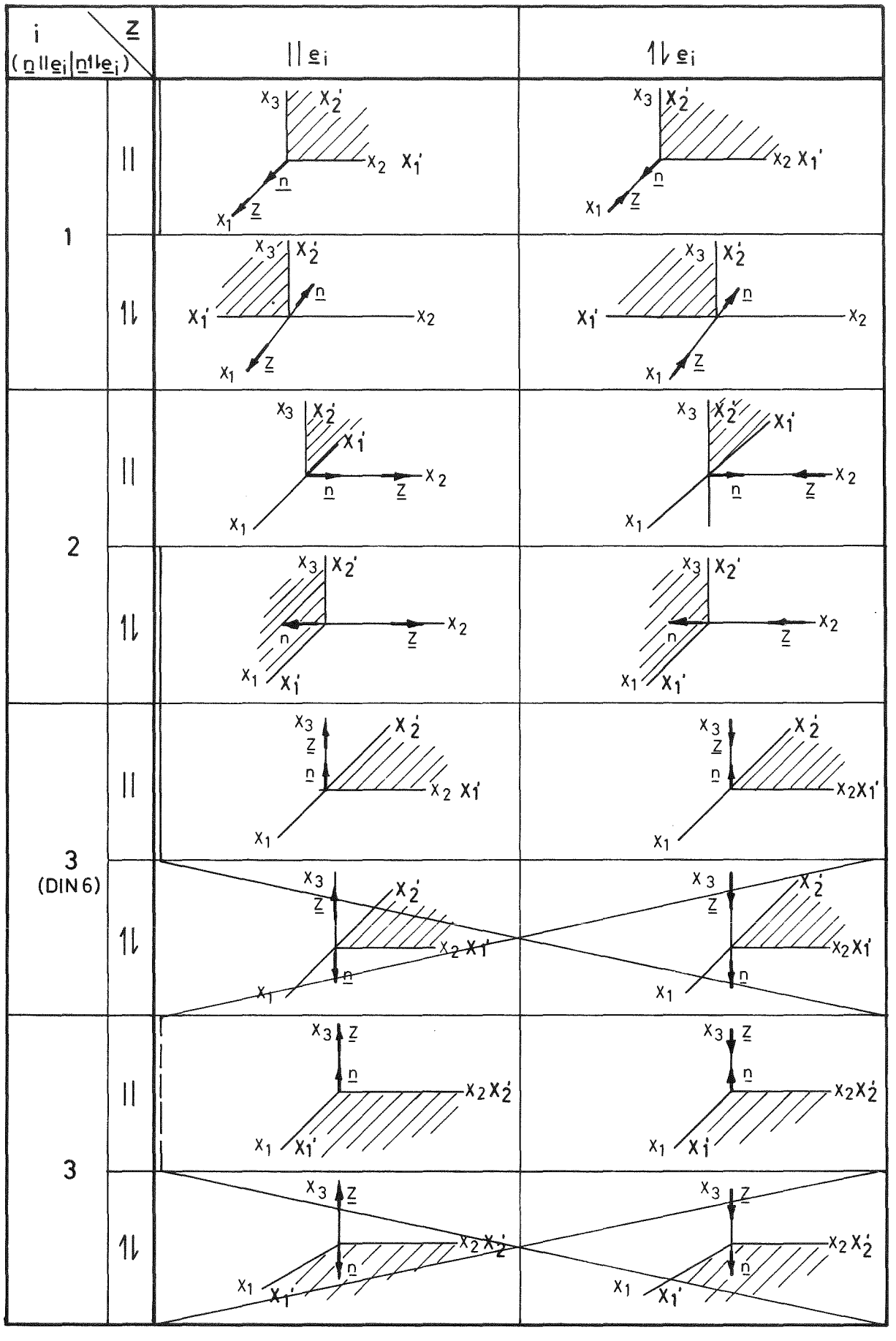


Abb. D.7: Anordnung von Bildebene und Projektionsrichtung mit resultierenden positivem Bildkoordinatenbereich

E. Bestimmung einer Orthonormalbasis

Die Beschaffung zweier orthonormierter Vektoren $\underline{a}$ und $\underline{b}$ in einer Ebene senkrecht zu $\underline{n}$ erfolgt nach dem E. Schmidt'schen Orthogonalisierungsverfahren. In einer für die Rechneranwendung aufbereiteten Form gilt

$$\underline{a} = \frac{\underline{c}}{|\underline{c}|} \qquad \text{und} \qquad \underline{b} = \frac{\underline{d}}{|\underline{d}|} \;,$$

wobei sich die Komponenten c_1, c_2 und c_3 bzw. d_1, d_2 und d_3 aus folgenden Beziehungen bestimmen:

$c_i = -n_j \cdot n_i$	$d_i = -n_k \cdot n_i$
$c_j = n_i^2$	$d_j = -n_k \cdot n_j$
$c_k = 0$	$d_k = n_i^2 + n_j^2$
$\lvert\underline{c}\rvert = n_i \sqrt{(n_i^2 + n_j^2)}$	$\lvert\underline{d}\rvert = \sqrt{n_k^2 (n_i^2 + n_j^2) + (n_i^2 + n_j^2)^2}$

Um die Entstehung eines Nullvektors zu vermeiden, darf die n_i-Komponente der Ebenennormale nicht gleich Null sein. Durch zyklisches Vertauschen der Indizes wird diese Möglichkeit ausgeschlossen:

	i, j, k
$n_1 \neq 0$	1, 2, 3
$n_2 \neq 0$	2, 3, 1
$n_3 \neq 0$	3, 1, 2

F. Analytische Bestimmung der zur Liniendarstellung erforderlichen Kanten und Punkte

F1. Durchdringungskurven

F1.1 Durchdringung Ebene$_1$ - Ebene$_2$

Der Verlauf der Schnittgeraden, die orthogonal zu den beiden Normalvektoren $\underline{n}_{E1}$ und $\underline{n}_{E2}$ verläuft, ergibt sich nach /61/ zu

$$\underline{X}(u) = \underline{p}_{E1} + \frac{(\underline{p}_{E1} - \underline{p}_{E2})\underline{n}_{E2}}{(\underline{n}_{E1} \times \underline{n}_{E2})^2}\left[(\underline{n}_{E1}\underline{n}_{E2})\,\underline{n}_{E1} - (\underline{n}_{E1}\underline{n}_{E2})\,\underline{n}_{E2}\right] + u(\underline{n}_{E1} \times \underline{n}_{E2}) \quad (f1)$$

F1.2 Durchdringung Ebene - Kugel

Die Schnittkurve zwischen Ebene und Kugel ist ein Kreis, dessen Gleichung sich leicht bestimmen läßt zu

$$\underline{X}(u) = \underline{m}_{Ku} + ((\underline{p}_E - \underline{m}_{Ku})\underline{n}_E)\underline{n}_E + r\,(\underline{a}\cos u + \underline{b}\sin u), \quad (\underline{a}, \underline{b} \perp \underline{n}_E) \quad (f2)$$

wobei

$$r = \sqrt{r_{Ku}^2 - ((\underline{p}_E - \underline{m}_{Ku})\underline{n}_E)^2}$$

ist.

F1.3 Durchdringung Ebene-Zylinder

Zur Berechnung des Verlaufes der Durchdringungskurve wird die parametrische Form der Zylindergleichung (6) in die vektorielle Ebenengleichung eingesetzt und so eine Beziehung

$$v(u) = \frac{1}{\underline{n}_E\underline{n}_z}\left[(\underline{p}_E - \underline{m}_z)\underline{n}_E - r_z\,(\underline{a}\,\underline{n}_E\cos u + \underline{b}\,\underline{n}_E\sin u)\right]$$

gefunden. Mit dieser Bedingung für v gilt

$$\underline{x}(u) = \underline{m}_z + \underline{n}_z \frac{((\underline{p}_E - \underline{m}_z)\underline{n}_E)}{(\underline{n}_E \underline{n}_z)} + \left(\underline{a} - \frac{(\underline{a}\,\underline{n}_E)}{(\underline{n}_E \underline{n}_z)} \underline{n}_z\right) \cos u + \left(\underline{b} - \frac{(\underline{b}\,\underline{n}_E)}{(\underline{n}_E \underline{n}_z)} \underline{n}_z\right) \sin u, \quad (f3)$$

$$(\underline{a}, \underline{b} \perp \underline{n}_z).$$

Für den Fall, daß Zylinderachse und Ebenennormale senkrecht zueinander stehen, wird

$$\underline{n}_E \, \underline{n}_Z = 0,$$

sodaß ein spezieller Ansatz erfolgt, den Abb. F.1 verdeutlicht.

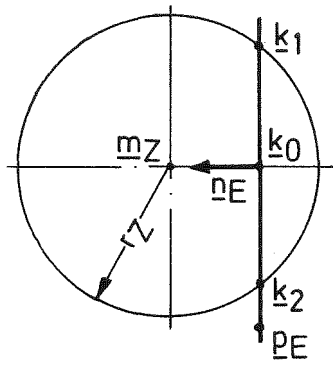


Abb. F.1: Schnitt eines Zylinders mit achsenparalleler Ebene

Durch die Punkte $\underline{k}_1$ und $\underline{k}_2$ ergeben sich Geraden

$$\underline{x}_i\,(u) = \underline{k}_i + u\,\underline{n}_Z, (i = 1,2),$$

die parallel zu $\underline{n}_Z$ verlaufen. $\underline{k}_1$ und $\underline{k}_2$ bestimmen sich aus den Schnittpunkten $\underline{x}_K = \underline{x}_E$ eines Kreises um $\underline{k}_o$, senkrecht zu $\underline{n}_E$ und mit Radius r:

$$\underline{x}_K(w) = \underline{k}_o + r\,(\underline{a} \cos w + \underline{b} \sin w),$$

wobei

$$\underline{k}_o = \underline{m}_z - ((\underline{m}_z - \underline{p}_E)\underline{n}_E)\,\underline{n}_E,$$

$$r = \sqrt{r_z^2 - ((\underline{m}_z - \underline{p}_E)\underline{n}_E)^2}$$

und

$$\underline{a}, \underline{b} \perp \underline{n}_E$$

sind, mit der Ebene

$$(\underline{x}_E - \underline{m}_Z)\,\underline{n}_Z = 0.$$

Für den Parameter w gilt hieraus folgende Beziehung

$$w_i = -\arctan\left(\frac{\underline{a}\,\underline{n}_z}{\underline{b}\,\underline{n}_z}\right) + (i-1)\cdot\pi\,, \quad (i=1,2)$$

und damit

$$\underline{x}_i(u) = \underline{m}_z - ((\underline{m}_z - \underline{p}_E)\underline{n}_E)\underline{n}_E + \sqrt{r_z^2 - ((\underline{m}_z - \underline{p}_E)\underline{n}_E)^2}\cdot \qquad (f4)$$

$$\left(\underline{a}\cos w_i + \underline{b}\sin w_i\right) + u\,\underline{n}_z\,, \quad (i=1,2),$$

$$(\underline{a}, \underline{b} \perp \underline{n}_E).$$

F1.4 Durchdringung Ebene-Kegel

Die Abhängigkeit des Parameters v von u folgt durch Einsetzen von (8) in (1), wodurch sich für

$$\underline{x}(u) = \underline{p}_{Ke} + \frac{((\underline{p}_E - \underline{p}_{Ke})\underline{n}_E)\left[(\underline{m}_{Ke} - \underline{p}_{Ke}) + r_{Ke}\left(\underline{a}\cos u + \underline{b}\sin u\right)\right]}{(\underline{m}_{Ke} - \underline{p}_{Ke})\underline{n}_E + r_{Ke}\left(\underline{a}\,\underline{n}_E\cos u + \underline{b}\,\underline{n}_E\sin u\right)}\,, \qquad (f5)$$

$$(\underline{a}, \underline{b} \perp (\underline{m}_{Ke} - \underline{p}_{Ke}))$$

ergibt.

F1.5 Durchdringung Kugel$_1$ - Kugel$_2$

Die Schnittkurve $\underline{x}_{Ku1} = \underline{x}_{Ku2}$ zweier Kugeln ist ein Kreis, für den nach Abb. F.2 folgende geometrische Überlegungen gelten:

$$\underline{x}_K(u) = \underline{p}_E + r\,(\underline{a}\cos u + \underline{b}\sin u),$$

$$\underline{p}_E = \underline{m}_{Ku1} + t\,(\underline{m}_{Ku2} - \underline{m}_{Ku1})$$

und

$\underline{a}$ und $\underline{b}$ senkrecht zu $(\underline{m}_{Ku1} - \underline{m}_{Ku2})$.

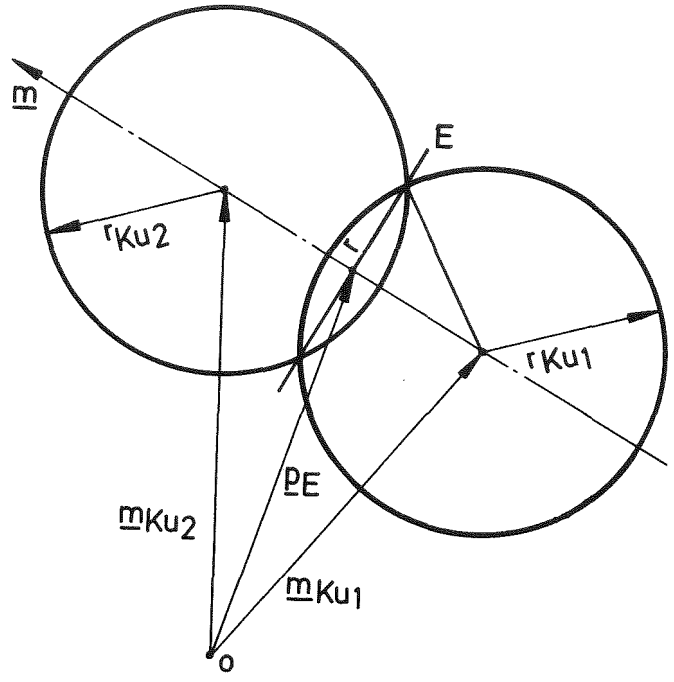


Abb. F.2: Geometrie der Kugel-Kugel-Durchdringung

Setzt man diese Beziehung in die durch Differenzbildung gewonnene Aussage

$$2\,\underline{x}\,(\underline{m}_{Ku2} - \underline{m}_{Ku1}) + \underline{m}^2_{Ku1} - \underline{m}^2_{Ku2} - r^2_{Ku1} + r^2_{Ku2} = 0$$

ein, so läßt sich t und damit $\underline{p}_E$ berechnen:

$$\underline{p}_E = \underline{m}_{Ku1} + (\underline{m}_{Ku2} - \underline{m}_{Ku1}) \frac{(\underline{m}_{Ku2} - \underline{m}_{Ku1})^2 + r^2_{Ku1} - r^2_{Ku2}}{2\,(\underline{m}_{Ku2} - \underline{m}_{Ku1})^2}$$

Mit

$$r = \sqrt{r^2_{Ku1} - (\underline{p}_E - \underline{m}_{Ku1})^2} \qquad \text{und} \qquad \underline{m} = \underline{m}_{Ku2} - \underline{m}_{Ku1}$$

hat der Schnittkreis dann folgenden Verlauf:

$$\underline{x}(u) = \underline{m}_{Ku1} + \underline{m}\,\frac{\underline{m}^2 + r^2_{Ku1} - r^2_{Ku2}}{2\,\underline{m}^2} + \sqrt{r^2_{Ku1} - \frac{(\underline{m}^2 + r^2_{Ku1} - r^2_{Ku2})^2}{4\,\underline{m}^2}} \cdot \quad \text{(f6)}$$

$$(\underline{a}\cos u + \underline{b}\sin u),$$

$$(\underline{a}, \underline{b} \perp (\underline{m}_{Ku2} - \underline{m}_{Ku1})).$$

F1.6 Durchdringung Kugel-Zylinder

Die Schnittkurve von Kugel und Zylinder läßt sich aus den Gleichungen (3) und (6) gewinnen. Der auch für die folgenden Herleitungen gültige Weg soll an diesem Schnitt exemplarisch etwas ausführlicher dargelegt werden, da er im Gegensatz zu den bisher mehr anschaulichen Ableitungen steht.

Für die Kugel gilt nach (3)

$$\underline{x}^2_{Ku} - 2\,\underline{x}_{Ku}\,\underline{m}_{Ku} + \underline{m}^2_{Ku} - r^2_{Ku} = 0$$

In diese Beziehung wird beim Schnitt $\underline{x}_{Ku} = \underline{x}_Z = \underline{x}$ die quadrierte Form von Gleichungen (6) eingesetzt:

$$\underline{x}_z^2 = \left(\left(\underline{m}_z + \underline{r}_a \cos u\right) + \left(\underline{r}_b \sin u + v\,\underline{n}_z\right)\right)^2$$

$$\underline{x}_z^2 = v^2\underline{n}_z^2 + 2v\left(\underline{r}_a\underline{n}_z\cos u + \underline{r}_b\underline{n}_z\sin u + \underline{m}_z\underline{n}_z\right)$$

$$+ \underline{m}_z^2 + 2\,\underline{m}_z\underline{r}_a\cos u + 2\,\underline{m}_z\underline{r}_b\sin u$$

$$+ \underline{r}_a^2\cos^2 u + \underline{r}_b^2\sin^2 u + 2\,\underline{r}_a\underline{r}_b\sin u\cos u$$

Um durch diese Gleichung auch den Fall elliptischer Zylinder abdecken zu können, wird die Möglichkeit $|\underline{r}_a| \neq |\underline{r}_b|$ zugelassen.
Für v ergibt sich daraus folgende quadratische Gleichung

$$v^2\underline{n}_z^2 + 2v\left[\underline{r}_a\underline{n}_z\cos u + \underline{r}_b\underline{n}_z\sin u + \underline{n}_z\left(\underline{m}_z - \underline{m}_{Ku}\right)\right]$$

$$+\left(\underline{m}_z - \underline{m}_{Ku}\right)^2 + \underline{r}_a^2\cos^2 u + \underline{r}_b^2\sin^2 u - r^2_{Ku} + 2\,\underline{r}_b\left(\underline{m}_z - \underline{m}_{Ku}\right)\sin u$$

$$+ 2\,\underline{r}_a\left(\underline{m}_z - \underline{m}_{Ku}\right)\cos u + 2\,\underline{r}_a\underline{r}_b\sin u\cos u = 0.$$

Für die beiden Wurzeln von v aus

$$c_2\,v^2 + c_1\,v + c_o = 0$$

gilt

$$v_{1,2} = \frac{1}{2C_2}\left(-C_1 \pm \sqrt{C_1^2 - 4C_oC_2}\right)$$

mit

$$C_1^2 = 4\Big[(\underline{r}_a \underline{n}_z)^2 \cos^2 u + 2(\underline{r}_a \underline{n}_z)(\underline{r}_b \underline{n}_z)\sin u \cos u + (\underline{r}_b \underline{n}_z)^2 \sin^2 u$$

$$+2(\underline{r}_a \underline{n}_z)(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku}))\cos u + 2(\underline{r}_b \underline{n}_z)(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku}))\sin u$$

$$+(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku}))^2\Big]$$

und

$$C_1^2 - 4C_0C_2 = 4\Big[(\underline{r}_a \underline{n}_z)^2 \cos^2 u + 2(\underline{r}_a \underline{n}_z)(\underline{r}_b \underline{n}_z)\sin u \cos u + (\underline{r}_b \underline{n}_z)^2 \sin^2 u$$

$$+2(\underline{r}_a \underline{n}_z)(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku}))\cos u + 2(\underline{r}_b \underline{n}_z)(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku}))\sin u$$

$$+(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku}))^2 - \underline{n}_z^2(\underline{m}_z - \underline{m}_{Ku})^2 - \underline{n}_z^2 \underline{r}_a^2 \cos^2 u$$

$$-\underline{n}_z^2 \underline{r}_b^2 \sin^2 u + \underline{n}_z^2 r_{Ku}^2 - 2\underline{n}_z^2(\underline{r}_b(\underline{m}_z - \underline{m}_{Ku}))\sin u$$

$$-2\underline{n}_z^2(\underline{r}_a(\underline{m}_z - \underline{m}_{Ku}))\cos u - 2\underline{n}_z^2(\underline{r}_a \underline{r}_b)\sin u \cos u\Big]$$

Durch Zusammenfassung ergibt sich:

$$C_1^2 - 4C_0C_2 = 4\Big\{\cos^2 u\Big[(\underline{r}_a \underline{n}_z)^2 - \underline{n}_z^2 \underline{r}_a^2\Big] + \sin^2 u\Big[(\underline{r}_b \underline{n}_z)^2 - \underline{n}_z^2 \underline{r}_b^2\Big]$$

$$+2\sin u \cos u\Big[(\underline{r}_a \underline{n}_z)(\underline{r}_b \underline{n}_z) - \underline{n}_z^2(\underline{r}_a \underline{r}_b)\Big]$$

$$+2\cos u\Big[(\underline{r}_a \underline{n}_z)(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku})) - \underline{n}_z^2(\underline{r}_a(\underline{m}_z - \underline{m}_{Ku}))\Big]$$

$$+2\sin u\Big[(\underline{r}_b \underline{n}_z)(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku})) - \underline{n}_z^2(\underline{r}_b(\underline{m}_z - \underline{m}_{Ku}))\Big]$$

$$+\Big[(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku}))^2 - \underline{n}_z^2(\underline{m}_z - \underline{m}_{Ku})^2\Big]$$

$$+\underline{n}_z^2 r_{Ku}^2$$

Mit Hilfe der Lagrange Identität

$$(\underline{a} \times \underline{b})(\underline{c} \times \underline{d}) = (\underline{a}\,\underline{c})(\underline{b}\,\underline{d}) - (\underline{b}\,\underline{c})(\underline{a}\,\underline{d})$$

lassen sich folgende Abkürzungen formulieren:

$$LA = \underline{n}_z^2\,\underline{r}_a^2 - (\underline{r}_a\,\underline{n}_z)^2 = (\underline{n}_z \times \underline{r}_a)^2,$$

$$LB = \underline{n}_z^2\,\underline{r}_b^2 - (\underline{r}_b\,\underline{n}_z)^2 = (\underline{n}_z \times \underline{r}_b)^2,$$

$$LAB = \underline{n}_z^2(\underline{r}_a\,\underline{r}_b) - (\underline{r}_a\,\underline{n}_z)(\underline{r}_b\,\underline{n}_z) = (\underline{n}_z \times \underline{r}_a)(\underline{n}_z \times \underline{r}_b),$$

$$LAM = \underline{n}_z^2(\underline{r}_a(\underline{m}_z - \underline{m}_{Ku})) - (\underline{r}_a\,\underline{n}_z)(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku})) = (\underline{n}_z \times \underline{r}_a)(\underline{n}_z \times (\underline{m}_z - \underline{m}_{Ku})),$$

$$LBM = \underline{n}_z^2(\underline{r}_b(\underline{m}_z - \underline{m}_{Ku})) - (\underline{r}_b\,\underline{n}_z)(\underline{n}_z(\underline{m}_z - \underline{m}_{Ku})) = (\underline{n}_z \times \underline{r}_b)(\underline{n}_z \times (\underline{m}_z - \underline{m}_{Ku}))$$

und

$$LM = \underline{n}_z^2(\underline{m}_z - \underline{m}_{Ku})^2 - (\underline{n}_z(\underline{m}_z - \underline{m}_{Ku}))^2 = (\underline{n}_z \times (\underline{m}_z - \underline{m}_{Ku}))^2$$

Unter Anwendung dieser Beziehung ergibt sich für die Schnittkurve zwischen Kugel und Zylinder:

$$\underline{X}(u) = \underline{m}_z + \underline{r}_a \cos u + \underline{r}_b \sin u$$

$$- \frac{\underline{n}_z}{\underline{n}_z^2}\Big[(\underline{r}_a\,\underline{n}_z)\cos u + (\underline{r}_b\,\underline{n}_z)\sin u + \underline{n}_z(\underline{m}_z - \underline{m}_{Ku})$$

$$\mp \sqrt{-LA\cos^2 u - LB\sin^2 u - 2\,LAB\sin u\cos u}$$

$$\overline{- 2\,LAM\cos u - 2\,LBM\sin u - LM + \underline{n}_z^2\,r_{Ku}^2}\Big]$$

Weitere Vereinfachungen durch Umrechnung trigonometrischer Funktionen führen auf

$$\underline{X}(u) = \underline{m}_z + \underline{r}_a \cos u + \underline{r}_b \sin u$$
$$- \frac{\underline{n}_z}{n_z^2} \Big[(\underline{r}_a \underline{n}_z) \cos u + (\underline{r}_b \underline{n}_z) \sin u + \underline{n}_z (\underline{m}_z - \underline{m}_{Ku})$$
$$\mp \sqrt{\tfrac{1}{2}(LB - LA) \cos 2u - LAB \sin 2u}$$
$$\overline{- 2\,LAM \cos u - 2\,LBM \sin u - \tfrac{1}{2}(LA + LB) - LM + \underline{n}_z^2 r_{Ku}^2} \Big]$$

Diese allgemeine Form der Schnittgleichung läßt sich durch folgende Bedingungen für den Kreiszylinder vereinfachen:

$$\underline{r}_a = r_z \underline{a}, \quad \underline{r}_b = r_z \underline{b},$$

$$LA = LB = r_z^2,$$

$$LAB = 0,$$

$$\underline{n}_z^2 = 1 \quad \text{und}$$

$$(\underline{a}\,\underline{n}_z), (\underline{b}\,\underline{n}_z) = 0,$$

$$\underline{X}(u) = \underline{m}_z + r_z (\underline{a} \cos u + \underline{b} \sin u) \qquad (f7)$$
$$- \underline{n}_z \Big[(\underline{n}_z (\underline{m}_z - \underline{m}_{Ku})) \mp \sqrt{r_{Ku}^2 - r_z^2 - LM - 2\,LAM \cos u - 2\,LBM \sin u} \Big],$$
$$(\underline{a}, \underline{b} \perp \underline{n}_z).$$

Fl. 7 Durchdringung Kugel-Kegel

Die Bestimmung von v (u) erfolgt aus der Anwendung der Gleichungen (3) und (8), wobei zunächst von der allgemeinen Kegelgleichung $|\underline{r}_a| \neq |\underline{r}_b|$ ausgegangen wird.

$$v^2 \left[(\underline{m}_{Ke} - \underline{p}_{Ke})^2 + 2((\underline{m}_{Ke} - \underline{p}_{Ke})\underline{r}_a)\cos u + 2((\underline{m}_{Ke} - \underline{p}_{Ke})\underline{r}_b)\sin u \right.$$

$$\left. + 2(\underline{r}_a \underline{r}_b)\sin u \cos u + \underline{r}_a^2 \cos^2 u + \underline{r}_b^2 \sin^2 u \right]$$

$$2v \left[\underline{p}_{Ke}(\underline{m}_{Ke} - \underline{p}_{Ke}) - (\underline{m}_{Ke} - \underline{m}_{Ku}) \right.$$

$$\left. + ((\underline{p}_{Ke}\underline{r}_a) - (\underline{m}_{Ku}\underline{r}_a))\cos u + ((\underline{p}_{Ke}\underline{r}_b)(\underline{m}_{Ku}\underline{r}_b))\sin u \right]$$

$$+ (\underline{p}_{Ke} - \underline{m}_{Ku})^2 - r_{Ku}^2 = 0$$

Nach Auflösung dieser Beziehung für v und Einführung folgender Abkürzungen

$$LA = \underline{r}_a(\underline{p}_{Ke} - \underline{m}_{Ku}) - \underline{r}_a^2(\underline{p}_{Ke} - \underline{m}_{Ku}) = (\underline{r}_a \times (\underline{p}_{Ke} - \underline{m}_{Ku}))(\underline{p}_{Ke} - \underline{m}_{Ku}) \times \underline{r}_a),$$

$$LB = \underline{r}_b(\underline{p}_{Ke} - \underline{m}_{Ku}) - \underline{r}_b^2(\underline{p}_{Ke} - \underline{m}_{Ku}) = (\underline{r}_b \times (\underline{p}_{Ke} - \underline{m}_{Ku}))(\underline{p}_{Ke} - \underline{m}_{Ku}) \times \underline{r}_b),$$

$$LAB = (\underline{r}_a(\underline{p}_{Ke} - \underline{m}_{Ku}))(\underline{r}_b(\underline{p}_{Ke} - \underline{m}_{Ku})) - (\underline{r}_a \underline{r}_b)(\underline{p}_{Ke} - \underline{m}_{Ku})^2 = (\underline{r}_a \times (\underline{p}_{Ke} - \underline{m}_{Ku}))((\underline{p}_{Ke} - \underline{m}_{Ku}) \times \underline{r}_b),$$

$$LAM = ((\underline{m}_{Ke} - \underline{p}_{Ke})(\underline{p}_{Ke} - \underline{m}_{Ku}))(\underline{r}_a(\underline{p}_{Ke} - \underline{m}_{Ku})) - ((\underline{m}_{Ke} - \underline{p}_{Ke})\underline{r}_a)(\underline{p}_{Ke} - \underline{m}_{Ku})^2$$

$$= ((\underline{m}_{Ke} - \underline{p}_{Ke}) \times (\underline{p}_{Ke} - \underline{m}_{Ku}))((\underline{p}_{Ke} - \underline{m}_{Ku}) \times \underline{r}_a),$$

$$LBM = ((\underline{m}_{Ke} - \underline{p}_{Ke})(\underline{p}_{Ke} - \underline{m}_{Ku}))(\underline{r}_b(\underline{p}_{Ke} - \underline{m}_{Ku})) - ((\underline{m}_{Ke} - \underline{p}_{Ke})\underline{r}_b)(\underline{p}_{Ke} - \underline{m}_{Ku})^2$$

$$= ((\underline{m}_{Ke} - \underline{p}_{Ke}) \times (\underline{p}_{Ke} - \underline{m}_{Ku}))(\underline{p}_{Ke} - \underline{m}_{Ku}) \times \underline{r}_b)$$

und

$$LM = ((\underline{m}_{Ke} - \underline{p}_{Ke})(\underline{p}_{Ke} - \underline{m}_{Ku}))^2 - (\underline{m}_{Ke} - \underline{p}_{Ke})^2(\underline{p}_{Ke} - \underline{m}_{Ku})^2$$

$$= ((\underline{m}_{Ke} - \underline{p}_{Ke}) \times (\underline{p}_{Ke} - \underline{m}_{Ku}))((\underline{p}_{Ke} - \underline{m}_{Ku}) \times (\underline{m}_{Ke} - \underline{p}_{Ke}))$$

ergibt sich

$$\underline{X}(u) = \underline{p}_{Ke} + \Big[(\underline{m}_{Ke} - \underline{p}_{Ke}) + \underline{r}_a \cos u + \underline{r}_b \sin u \Big] \cdot$$

$$\frac{-\Big[(\underline{m}_{Ke} - \underline{p}_{Ke})(\underline{p}_{Ke} - \underline{m}_{Ku}) + \underline{r}_a (\underline{p}_{Ke} - \underline{m}_{Ku}) \cos u \; \ldots}{\frac{1}{2}(\underline{r}_a^2 + \underline{r}_b^2) + (\underline{m}_{Ke} - \underline{p}_{Ke})^2 + 2(\underline{m}_{Ke} - \underline{p}_{Ke})\underline{r}_a \cos u}$$

$$\ldots \frac{\underline{r}_b (\underline{p}_{Ke} - \underline{m}_{Ku}) \sin u \Big] \pm \sqrt{R}}{+2(\underline{m}_{Ke} - \underline{p}_{Ke})\underline{r}_b \sin u + \underline{r}_a \underline{r}_b \sin 2u + \frac{1}{2}\cos 2u (\underline{r}_a^2 - \underline{r}_b^2)}$$

$$R = \tfrac{1}{2}(LA - LB + r_{Ku}^2(\underline{r}_a^2 - \underline{r}_b^2))\cos 2u + (LAB + r_{Ku}^2(\underline{r}_a \underline{r}_b))\sin 2u$$

$$+2(LAM + r_{Ku}^2(\underline{m}_{Ke} - \underline{p}_{Ke})\underline{r}_a)\cos u + 2(LBM + r_{Ku}^2(\underline{m}_{Ke} - \underline{p}_{Ke})\underline{r}_b)\sin u$$

$$LM + \tfrac{1}{2}(LA + LB) + r_{Ku}^2\Big((\underline{m}_{Ke} - \underline{p}_{Ke})^2 + \tfrac{1}{2}\underline{r}_a^2 + \tfrac{1}{2}\underline{r}_b^2\Big)$$

Durch folgende Aussagen für den Kreiskegel läßt sich die Gleichung vereinfachen:

$$\underline{r}_a = r_{Ke}\,\underline{a}\,, \quad \underline{r}_b = r_{Ke}\,\underline{b}\,,$$

$$\underline{a}\,(\underline{m}_{Ke} - \underline{p}_{Ke})\,, \quad \underline{b}\,(\underline{m}_{Ke} - \underline{p}_{Ke})\,, \quad \underline{a}\,\underline{b} = 0$$

$$\underline{x}(u) = \underline{p}_{Ke} + \Big[(\underline{m}_{Ke} - \underline{p}_{Ke}) + r_{Ke}(\underline{a}\cos u + \underline{b}\sin u)\Big]. \qquad (f8)$$

$$\frac{-\Big[(\underline{m}_{Ke} - \underline{p}_{Ke})(\underline{p}_{Ke} - \underline{m}_{Ku}) + r_{Ke}\,(\underline{a}(\underline{p}_{Ke} - \underline{m}_{Ku})\cos u + \underline{b}(\underline{p}_{Ke} - \underline{m}_{Ku})\sin u)\Big] \pm \sqrt{R}}{r_{Ke}^2 + (\underline{m}_{Ke} - \underline{p}_{Ke})^2}$$

$$R = \tfrac{1}{2}(LA - LB)\cos 2u + LAB \sin 2u$$

$$+2\,LAM\cos u + 2\,LBM\sin u$$

$$+LM + \tfrac{1}{2}(LA + LB) + r_{Ku}^2\Big((\underline{m}_{Ke} - \underline{p}_{Ke})^2 + r_{Ke}^2\Big),$$

$$(\underline{a}, \underline{b} \perp (\underline{m}_{Ke} - \underline{p}_{Ke})).$$

F1.8 Durchdringung Zylinder$_1$ - Zylinder$_2$

Zur Berechnung dieser Durchdringungskurve werden die Zylindergleichungen (5) und (6) herangezogen. Zur Beseitigung der Vektorprodukte wird (5) quadriert, sodaß gilt:

$$r^2_Z \, \underline{n}^2_Z = ((\underline{x} - \underline{m}_Z) \times \underline{n}_Z)^2 = (\underline{x} \times \underline{n}_Z)^2 - 2(\underline{x} \times \underline{n}_Z)\,(\underline{m}_Z \times \underline{n}_Z) + (\underline{m}_Z \times \underline{n}_Z)^2$$

Dieser Ausdruck kann mittels der Lagrange Identität in eine Skalarproduktform umgesetzt werden:

$$r^2_Z \, \underline{n}^2_Z = \underline{x}^2 \, \underline{n}^2_Z - (\underline{x}\,\underline{n}_Z)^2 - 2[(\underline{x}\,\underline{m}_Z)\,\underline{n}^2_Z - (\underline{x}\,\underline{n}_Z)(\underline{n}_Z\underline{m}_Z)] + \underline{m}^2_Z \, \underline{n}^2_Z - (\underline{m}_Z\underline{n}_Z)^2$$

In diese Gleichung wird die parametrische Darstellung des zweiten Zylinders in allgemeiner Form eingesetzt. Nach einigen Schritten wird mit folgenden Abkürzungen

$$LA = \underline{n}^2_{z1}\,\underline{r}_a^2 - (\underline{r}_a\,\underline{n}_{z1})^2 = (\underline{r}_a \times \underline{n}_{z1})^2,$$

$$LB = \underline{n}^2_{z1}\,\underline{r}_b^2 - (\underline{r}_b\,\underline{n}_{z1})^2 = (\underline{r}_b \times \underline{n}_{z1})^2,$$

$$LAB = \underline{n}^2_{z1}\,(\underline{r}_a\,\underline{r}_b) - (\underline{r}_a\,\underline{n}_{z1})(\underline{r}_b\,\underline{n}_{z1}) = (\underline{n}_{z1} \times \underline{r}_a)(\underline{n}_{z1} \times \underline{r}_b),$$

$$LAN = \underline{n}^2_{z1}(\underline{r}_a\,\underline{n}_{z2}) - (\underline{r}_a\,\underline{n}_{z1})(\underline{n}_{z1}\,\underline{n}_{z2}) = (\underline{n}_{z1} \times \underline{r}_a)(\underline{n}_{z1} \times \underline{n}_{z2}),$$

$$LBN = \underline{n}^2_{z1}(\underline{r}_b\,\underline{n}_{z2}) - (\underline{r}_b\,\underline{n}_{z1})(\underline{n}_{z1}\,\underline{n}_{z2}) = (\underline{n}_{z1} \times \underline{r}_b)(\underline{n}_{z1} \times \underline{n}_{z2}),$$

$$LM = \underline{n}^2_{z1}(\underline{m}_{z2} - \underline{m}_{z1})^2 - ((\underline{m}_{z2} - \underline{m}_{z1})\,\underline{n}_{z1})^2 = (\underline{n}_{z1} \times (\underline{m}_{z2} - \underline{m}_{z1}))^2,$$

$$LMN = \underline{n}^2_{z1}(\underline{n}_{z2}(\underline{m}_{z2} - \underline{m}_{z1})) - (\underline{n}_{z2}\,\underline{n}_{z1})(\underline{n}_{z1}(\underline{m}_{z2} - \underline{m}_{z1})) = (\underline{n}_{z1} \times \underline{n}_{z2})(\underline{n}_{z1} \times (\underline{m}_{z2} - \underline{m}_{z1})),$$

$$LN = \underline{n}^2_{z2}\,\underline{n}^2_{z1} - (\underline{n}_{z2}\,\underline{n}_{z1})^2 = (\underline{n}_{z2} \times \underline{n}_{z1})^2,$$

$$LAM = \underline{n}^2_{z1}(\underline{r}_a(\underline{m}_{z2} - \underline{m}_{z1})) - (\underline{r}_a\,\underline{n}_{z1})(\underline{n}_{z1}(\underline{m}_{z2} - \underline{m}_{z1})) = (\underline{n}_{z1} \times \underline{r}_a)(\underline{n}_{z1} \times (\underline{m}_{z2} - \underline{m}_{z1}))$$

und

$$LBM = \underline{n}^2_{z1}(\underline{r}_b(\underline{m}_{z2} - \underline{m}_{z1})) - (\underline{r}_b\,\underline{n}_{z1})(\underline{n}_{z1}(\underline{m}_{z2} - \underline{m}_{z1})) = (\underline{n}_{z1} \times \underline{r}_b)(\underline{n}_{z1} \times (\underline{m}_{z2} - \underline{m}_{z1}))$$

die Beziehung für die Schnittkurve gewonnen:

$$\underline{X}(u) = \underline{m}_{z2} + r_{z2}(\underline{a}\cos u + \underline{b}\sin u) - \frac{n_{z2}}{LN}\left[LAN\cos u + LBN\sin u + LMN \mp \sqrt{R}\right] \tag{f9}$$

$$R = \frac{1}{2}(LAN^2 - LBN^2 - LN(LA - LB))\cos 2u$$

$$+ (LBN \cdot LAN - LN \cdot LAB)\sin 2u$$

$$+ 2(LMN \cdot LAN - LN \cdot LAM)\cos u$$

$$+ 2(LMN \cdot LBN - LN \cdot LBM)\sin u$$

$$+ LMN^2 + \frac{1}{2}(LAN^2 + LBN^2) - LN(LM + \frac{1}{2}(LA + LB) - r_{z1}^2 \underline{n}_{z1}^2),$$

$$(\underline{a}, \underline{b} \perp \underline{n}_{z_2})$$

Die Annahme eines Kreiszylinders führt zu folgenden Bedingungen:

$$\underline{r}_a = r_{Z_2}\,\underline{a} \quad \text{und} \quad \underline{r}_b = r_{Z_2}\,\underline{b}.$$

Wenn die Achsen der Zylinder parallel verlaufen, wird

$$LN = (\underline{n}_{Z_2} \times \underline{n}_{Z_2})^2 = 0.$$

Für diesen Sonderfall ergeben sich als Durchdringungskanten zwei Geraden

$$\underline{x}_i(u) = \underline{k}_i + u\,\underline{n}_{Z1}\,, \quad (i = 1,2),$$

die in einer Ebene senkrecht zu $(\underline{m}_{Z_2} - \underline{m}^*_{Z_1})$ liegen. Um die Lage des Punktes $m^*_{Z_1}$ zu finden, der auf der Achse des ersten Zylinders in einer Ebene senkrecht zu $\underline{n}_{Z1} = \underline{n}_{Z_2}$ und durch $\underline{m}_{Z_2}$ liegen soll, wird die Gerade

$$\underline{m}^*_{Z_1} = \underline{m}_{Z1} + t_o\,\underline{n}_{Z1}$$

mit der Ebene

$$(\underline{m}^*_{Z_1} - \underline{m}_{Z_2})\,\underline{n}_{Z1} = 0$$

geschnitten, so daß

$$\underline{m}^*_{Z_1} = \underline{m}_{Z1} + ((\underline{m}_{Z_2} - \underline{m}_{Z1})\,\underline{n}_{Z1})\,\underline{n}_{Z1}$$

ist.

Wenn wir dann den Schnitt durch zwei Kugeln als Hilfsflächen ausgeführt denken, so können wir in Analogie zur Kugel-Kugel Verschneidung (f6) schreiben:

$$\underline{k}_o = \underline{m}^*_{z1} + (\underline{m}_{z2} - \underline{m}^*_{z1}) \quad \frac{(\underline{m}_{z2} - \underline{m}^*_{z1})^2 + r_{z1}^2 - r_{z2}^2}{2\,(\underline{m}_{z2} - \underline{m}^*_{z1})^2}$$

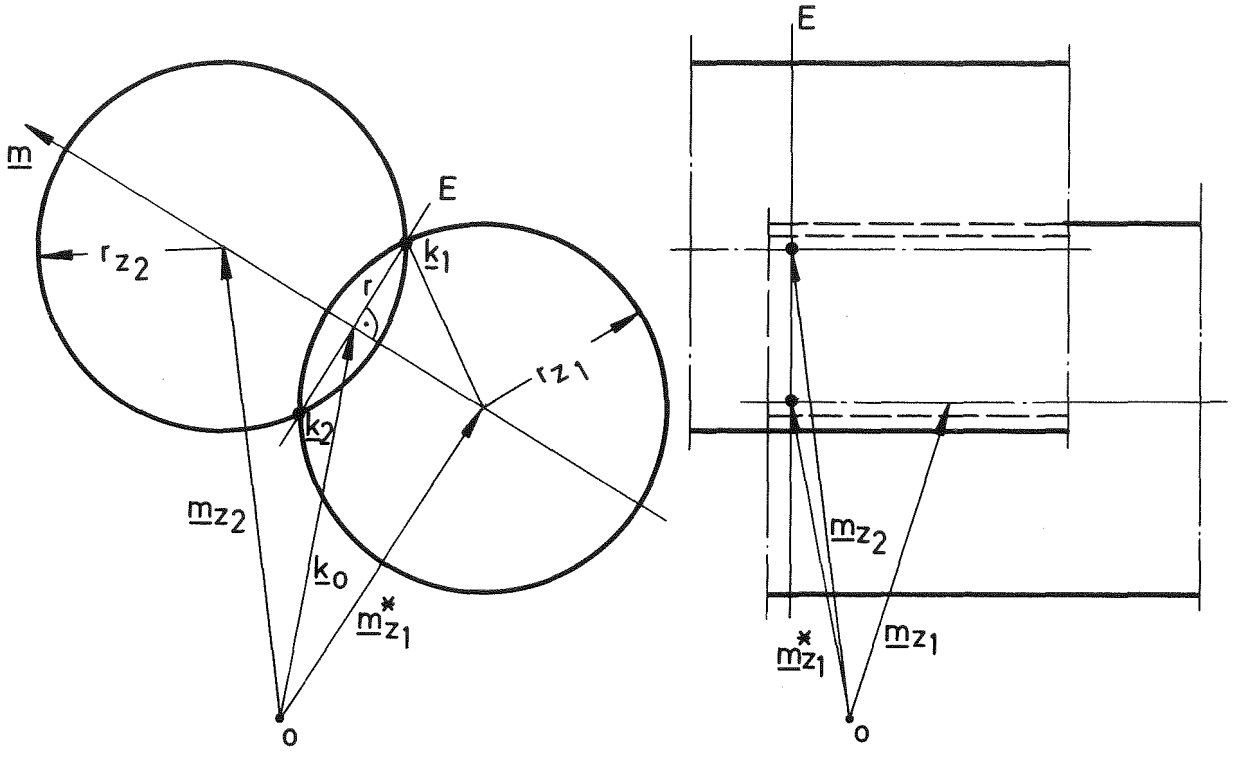


Abb. F.3: Durchdringung paralleler Zylinder

Mit dieser Beziehung kann die weitere Rechnung in Entsprechung zum Sonderfall des Ebenen-Zylinder-Schnitts ablaufen.
Es gilt

$$r = \sqrt{r_{t1}^2 - (\underline{k}_o - \underline{m}^*_{t1})^2} \,,$$

$$\underline{m} = \underline{m}_{t2} - \underline{m}^*_{t1}$$

und

$$\underline{a}, \underline{b} \perp (\underline{m}_{t2} - \underline{m}^*_{t1}) \,,$$

so daß

$$\underline{X}(u) = \underline{m}_{z1} + \underline{m}\,\frac{\underline{m}^2 + r_{z1}^2 - r_{z2}^2}{2\,\underline{m}^2} + \sqrt{r_{z1}^2 - \frac{(\underline{m}^2 + r_{z1}^2 - r_{z2}^2)^2}{4\,\underline{m}^2}}\;\cdot$$

$$(\underline{a}\cos w_i + \underline{b}\sin w_i) + u\cdot\underline{n}_{z1} \qquad \text{(f10)}$$

$$(\underline{a},\underline{b} \perp (\underline{m}_{z2} - \underline{m}^*_{z1}))$$

wird, wobei

$$w_i = -\arctan\left(\frac{\underline{a}\;\underline{n}_{z1}}{\underline{b}\;\underline{n}_{z1}}\right) + (i-1)\cdot\pi \quad , \quad (i=1,2).$$

F1.9 Durchdringung Zylinder-Kegel

Die Berechnung des Zylinder-Kegel-Schnittes geht von den Gleichungen (5) und (8) aus. Nach Einsetzen der parametrischen Kegelgleichung in die quadrierte vektorielle Zylindergleichung und einigen Umrechnungen ist v wiederum durch eine quadratische Gleichung gegeben, die mit folgenden Abkürzungen

$$LA = r_a^2\,\underline{n}_z^2 - (r_a\,\underline{n}_z)^2 = (r_a \times \underline{n}_z)^2,$$

$$LB = r_b^2\,\underline{n}_z^2 - (r_b\,\underline{n}_z)^2 = (r_b \times \underline{n}_z)^2,$$

$$LAB = \underline{n}_z^2\,(r_a\,r_b) - (\underline{n}_z r_a)(r_b\,\underline{n}_z) = (\underline{n}_z \times r_a)(\underline{n}_z \times r_b),$$

$$LAN = \underline{n}_z^2\,((\underline{m}_{Ke} - p_{Ke})\,r_a) - (\underline{n}_z(\underline{m}_{Ke} - p_{Ke}))(\underline{n}_z\,r_a) = (\underline{n}_z \times (\underline{m}_{Ke} - p_{Ke}))(\underline{n}_z \times r_a),$$

$$LBN = \underline{n}_z^2\,((\underline{m}_{Ke} - p_{Ke})\,r_b) - (\underline{n}_z(\underline{m}_{Ke} - p_{Ke}))(\underline{n}_z\,r_b) = (\underline{n}_z \times (\underline{m}_{Ke} - p_{Ke}))(\underline{n}_z \times r_b),$$

$$LM = \underline{n}_z^2\,(p_{Ke} - \underline{m}_z)^2 - (\underline{n}_z(p_{Ke} - \underline{m}_z))^2 - r_z^2\,\underline{n}_z^2 = (\underline{n}_z \times (p_{Ke} - \underline{m}_z))^2 - r_z^2\,\underline{n}_z^2,$$

$$LMN = \underline{n}_z^2\,((p_{Ke} - \underline{m}_z)(\underline{m}_{Ke} - p_{Ke})) - (\underline{n}_z(\underline{m}_{Ke} - p_{Ke}))((p_{Ke} - \underline{m}_z)\,\underline{n}_z) = (\underline{n}_z \times (p_{Ke} - \underline{m}_z))(\underline{n}_z \times (\underline{m}_{Ke} - p_{Ke})),$$

$$LN = (\underline{m}_{Ke} - p_{Ke})^2\,\underline{n}_z^2 - ((\underline{m}_{Ke} - p_{Ke})\,\underline{n}_z)^2 = ((\underline{m}_{Ke} - p_{Ke}) \times \underline{n}_z)^2,$$

$$LAM = \underline{n}_z^2\,((p_{Ke} - \underline{m}_z)\,r_a) - (\underline{n}_z\,r_a)((p_{Ke} - \underline{m}_z)\,\underline{n}_z) = (\underline{n}_z \times (p_{Ke} - \underline{m}_z))(\underline{n}_z \times r_a)$$

und

$$LBM = \underline{n}_z^2\,((p_{Ke} - \underline{m}_z)\,r_b) - (\underline{n}_z\,r_b)((p_{Ke} - \underline{m}_z)\,\underline{n}_z) = (\underline{n}_z \times (p_{Ke} - \underline{m}_z))(\underline{n}_z \times r_b)$$

zur Beziehung

$$\underline{x}(u) = \underline{p}_{Ke} + \left[(\underline{m}_{Ke} - \underline{p}_{Ke}) + r_{Ke}(\underline{a}\cos u + \underline{b}\sin u)\right] \cdot \qquad \text{(f11)}$$

$$\frac{-(LMN + LAM\cos u + LBM\sin u) \pm \sqrt{R}}{LN + \frac{1}{2}(LA+LB) + \frac{1}{2}(LA-LB)\cos 2u + LAB\sin 2u + 2LAN\cos u + 2LBN\sin u}$$

$$R = \tfrac{1}{2}(LAM^2 - LBM^2 - (LA-LB)\cdot LM)\cos 2u$$

$$+ (LAM\cdot LBM - LAB\cdot LM)\sin 2u$$

$$+ 2(LAM\cdot LMN - LAN\cdot LM)\cos u$$

$$+ 2(LBM\cdot LMN - LBN\cdot LM)\sin u$$

$$+ LMN^2 + \tfrac{1}{2}(LAM^2 + LBM^2) - (LN + \tfrac{1}{2}(LA+LB))\cdot LM,$$

$$(\underline{a}, \underline{b} \perp (\underline{m}_{Ke} - \underline{p}_{Ke}))$$

führt.

Fl.10 Durchdringung Kegel$_1$ - Kegel$_2$

Die Berechnung der gemeinsamen Lösung für zwei Kegel geht von der unterschiedlichen Darstellung in den Gleichungen (7) und (8) aus. Durch Index 2 wird der zweite in parametrischer Form beschriebene Schnittpartner gekennzeichnet.
Nach Einführung folgender Beziehungen

$$LA = r_a^2 + ((\underline{m}_{Ke1} - \underline{p}_{Ke1})\,\underline{r}_a)^2,$$

$$LB = r_b^2 + ((\underline{m}_{Ke1} - \underline{p}_{Ke1})\,\underline{r}_b)^2,$$

$$LAB = (\underline{r}_a\,\underline{r}_b) + ((\underline{m}_{Ke1} - \underline{p}_{Ke1})\,\underline{r}_a)((\underline{m}_{Ke1} - \underline{p}_{Ke1})\,\underline{r}_b),$$

$$LAN = ((\underline{m}_{Ke2} - \underline{p}_{Ke2})\,\underline{r}_a) + ((\underline{m}_{Ke1} - \underline{p}_{Ke1})(\underline{m}_{Ke2} - \underline{p}_{Ke2}))((\underline{m}_{Ke1} - \underline{p}_{Ke1})\,\underline{r}_a),$$

$$LBN = ((\underline{m}_{Ke2} - \underline{p}_{Ke2})\underline{\tau}_b) + ((\underline{m}_{Ke1} - \underline{p}_{Ke1})(\underline{m}_{Ke2} - \underline{p}_{Ke2}))((\underline{m}_{Ke1} - \underline{p}_{Ke1})\underline{\tau}_b),$$

$$LM = ((\underline{m}_{Ke1} - \underline{p}_{Ke1})(\underline{p}_{Ke2} - \underline{p}_{Ke1}))^2 - (\underline{m}_{Ke1} - \underline{p}_{Ke1})^2(\underline{p}_{Ke2} - \underline{p}_{Ke1})^2 \cos^2\delta,$$

$$LMN = ((\underline{m}_{Ke1} - \underline{p}_{Ke1})(\underline{m}_{Ke2} - \underline{p}_{Ke2}))((\underline{p}_{Ke2} - \underline{p}_{Ke1})(\underline{m}_{Ke1} - \underline{p}_{Ke1}))$$
$$- (\underline{m}_{Ke1} - \underline{p}_{Ke1})^2((\underline{m}_{Ke2} - \underline{p}_{Ke2})(\underline{p}_{Ke2} - \underline{p}_{Ke1}))\cos^2\delta,$$

$$LN = (\underline{m}_{Ke2} - \underline{p}_{Ke2})^2 + ((\underline{m}_{Ke1} - \underline{p}_{Ke1})(\underline{m}_{Ke2} - \underline{p}_{Ke2}))^2,$$

$$LAM = ((\underline{m}_{Ke1} - \underline{p}_{Ke1})\underline{\tau}_a)((\underline{p}_{Ke2} - \underline{p}_{Ke1})(\underline{m}_{Ke1} - \underline{p}_{Ke1}))$$
$$- (\underline{m}_{Ke1} - \underline{p}_{Ke1})(\underline{\tau}_a(\underline{p}_{Ke2} - \underline{p}_{Ke1}))\cos^2\delta$$

und

$$LBM = ((\underline{m}_{Ke1} - \underline{p}_{Ke1})\underline{\tau}_b)((\underline{p}_{Ke2} - \underline{p}_{Ke1})(\underline{m}_{Ke1} - \underline{p}_{Ke1}))$$
$$- (\underline{m}_{Ke1} - \underline{p}_{Ke1})(\underline{\tau}_b(\underline{p}_{Ke2} - \underline{p}_{Ke1}))\cos^2\delta$$

gilt

$$\underline{x}(u) = \underline{p}_{Ke2} + \left[(\underline{m}_{Ke2} - \underline{p}_{Ke2}) + \tau_{Ke2}(\underline{a}\cos u + \underline{b}\sin u)\right] \cdot \qquad \text{(f12)}$$

$$\frac{-(LMN + LAM\cos u + LBM\sin u) \pm \sqrt{R}}{LN + \frac{1}{2}(LA+LB) + \frac{1}{2}(LA-LB)\cos 2u + LAB\sin 2u + 2LAN\cos u + 2LBN\sin u}$$

$$R = \tfrac{1}{2}\,(LAM^2 - LBM^2 - (LA-LB)LM)\cos 2u$$

$$+ (LAM\cdot LBM - LAB\cdot LM)\sin 2u$$

$$+ 2(LAM\cdot LMN - LAN\cdot LM)\cdot\cos u$$

$$+ 2(LBM\cdot LMN - LBN\cdot LM)\cdot\sin u$$

$$+ LMN^2 + \tfrac{1}{2}(LAM^2 + LBM^2) - (LN + \tfrac{1}{2}(LA+LB))LM,$$

$$(\underline{a}, \underline{b} \perp (\underline{m}_{Ke2} - \underline{p}_{Ke2})).$$

F2. Umrißkurven

Zur Darstellung der Flächen sind die Kurven der wahren Umrisse der Flächen Kugel, Zylinder und Kegel zu bestimmen. Der wahre Umriß einer Fläche ist definiert als Ort der Punkte in denen ein Projektionsstrahl die Fläche tangiert. Bei Quadriken liegen diese Punkte in einer Ebene, der sogenannten Polarebene, die gegeben ist durch den Ort der Punkte $\underline{x}$, die zu den Schnittpunkten einer Geraden durch das Zentrum $\underline{Z}$ (Pol) mit der Quadrik ein Doppelverhältnis DV = -1 haben. Das Umrißproblem einer Quadrik ließe sich so auf den Schnitt mit der Polarebene

$$\underline{Z}^T \underline{\underline{C}}_H \underline{X} = 0$$

zurückführen.
Im Sinne einer effektiven Ausführung wird hier eine explizite Bestimmung vorgenommen.

F2.1 Umriß Kugel

Der wahre Umriß einer Kugel ist ein Kreis /63/:

$$\underline{x}(u) = \underline{k}_o + r\,(\underline{a}\cos u + \underline{b}\sin u).$$

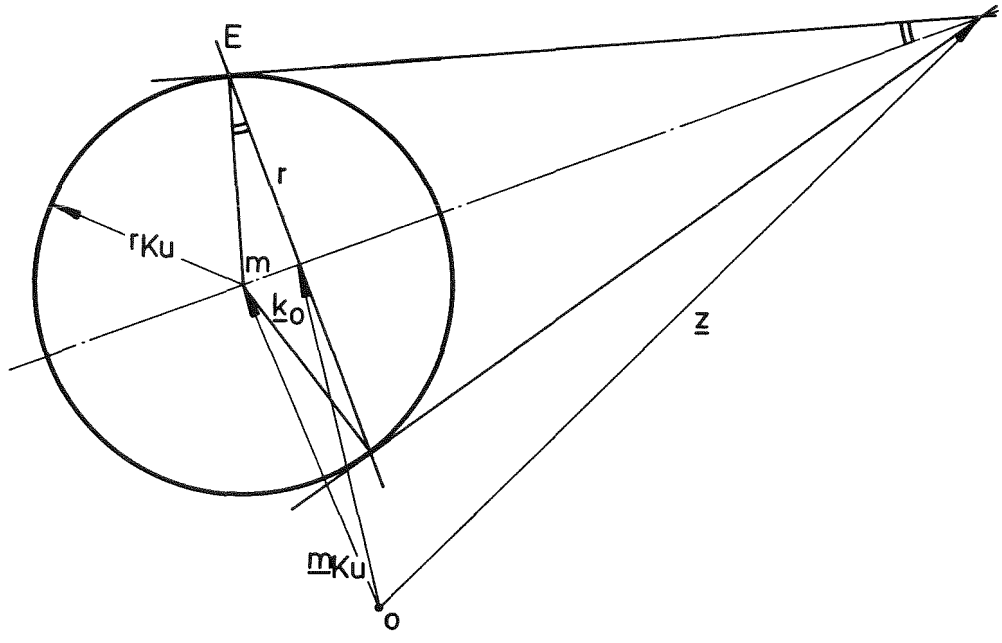


Abb. F.4: Wahrer Kugelumriß bei Zentralprojektion

Es gilt

$$\underline{k}_o = \underline{m}_{Ku} + m \frac{(\underline{z} - \underline{m}_{Ku})}{|\underline{z} - \underline{m}_{Ku}|} ,$$

wobei m leicht bestimmt wird zu

$$m = \frac{r_{Ku}^2}{|\underline{z} - \underline{m}_{Ku}|}$$

Der Radius für den Kreis des wahren Umriß ist

$$r = r_{Ku} \sqrt{1 - \frac{r_{Ku}^2}{(\underline{z} - \underline{m}_{Ku})^2}} ,$$

so daß sich ergibt

$$\underline{x}(u) = \underline{m}_{Ku} + \frac{r_{Ku}^2}{(\underline{z} - \underline{m}_{Ku})^2}(\underline{z} - \underline{m}_{Ku}) + r_{Ku} \sqrt{1 - \frac{r_{Ku}^2}{(\underline{z} - \underline{m}_{Ku})^2}} (\underline{a} \cos u + \underline{b} \sin u). \quad (f13)$$

Für den Fall der Parallelprojektion wird mit m = 0 und r = r_{Ku}:

$$\underline{x}(u) = \underline{m}_{Ku} + r_{Ku} (\underline{a} \cos u + \underline{b} \sin u). \quad (f14)$$

$\underline{a}$ und $\underline{b}$ sind orthonormierte Vektoren in der Ebene senkrecht zu $(\underline{z} - \underline{m}_{Ku})$ bzw. bei Parallelprojektion senkrecht zur Betrachtungsrichtung.

F2.2 Umriß Zylinder

Zur Bestimmung des wahren Umrisses des Zylinders, der durch die Berührungsgeraden

$$\underline{x}_i(u) = \underline{k}_i + u \cdot \underline{n}_Z \quad , (i = 1,2)$$

der Tangentialebenen durch das Projektionszentrum gegeben ist, wird eine Kugel mit Radius r_Z um den Punkt $\underline{m}_Z^*$ gelegt, der sich durch den Schnitt einer Ebene mit Normale $\underline{n}_Z$ und durch $\underline{z}$ mit der Zylinderachse ergibt:

$$\underline{m}_Z^* = \underline{m}_Z + ((\underline{z} - \underline{m}_Z)\,\underline{n}_Z)\,\underline{n}_Z$$

Die Schnittpunkte des Umrisses dieser Kugel mit der Ebene sind bestimmt nach (f13) als

$$\underline{k}_i = \underline{m}_z^* + \frac{r_z^2}{(\underline{z} - \underline{m}_z^*)^2}(\underline{z} - \underline{m}_z^*) + r_z\sqrt{1 - \frac{r_z^2}{(\underline{z} - \underline{m}_z^*)^2}}\,(\underline{a}\cos w_i + \underline{b}\sin w_i)$$

mit

$$w_i = -\arctan\left(\frac{\underline{a}\,\underline{n}_z}{\underline{b}\,\underline{n}_z}\right) + (i-1)\cdot\pi \quad , \quad (i = 1,2).$$

Für die Berührungsgeraden der Sehstrahlen gilt dann

$$\underline{x}_i(u) = \underline{m}_z^* + \frac{r_z^2}{(\underline{z} - \underline{m}_z^*)^2}(\underline{z} - \underline{m}_z^*) + r_z\sqrt{1 - \frac{r_z^2}{(\underline{z} - \underline{m}_z^*)^2}}\,(\underline{a}\cos w_i + \underline{b}\sin w_i) + u\cdot\underline{n}_z \quad , \quad (i = 1,2) \qquad \text{(f15)}$$

$$(\underline{a}, \underline{b} \perp (\underline{z} - \underline{m}_z^*)).$$

Bei Parallelprojektion geht diese Gleichung wieder über in die Form

$$\underline{x}_i(u) = \underline{m}_z + r_z(\underline{a}\cos w_i + \underline{b}\sin w_i) + u\cdot\underline{n}_z \quad , \quad (i = 1,2) \qquad \text{(f16)}$$

$$(\underline{a}, \underline{b} \perp \underline{z}).$$

F2.3 Umriß Kegel

Zur Berechnung der wahren Umrißgeraden des Kegels

$$\underline{x}_i(u) = \underline{p}_{Ke} + u\ (\underline{k}_i - \underline{p}_{Ke})\ ,\ (i = 1,2)$$

wird eine Ersatzkugel eingebracht, so daß die Kegelspitze $\underline{p}_{Ke}$ Pol dieser Kugel ist. Der Punkt $\underline{p}$ entsteht durch Schneiden des Projektionsstrahles durch $\underline{p}_{Ke}$ mit der Polarebene $E_{\underline{p}_{Ke}}$. $\underline{p}$ stellt nun seinerseits den Pol einer Polarebene $E_{\underline{p}}$ der Kugel, in der $\underline{p}_{Ke}$ liegt, dar. Die Punkte $\underline{k}_i$ der Umrißkanten liegen nun in den Schnittpunkten der beiden wahren Umrisse der gedachten Kugel bezüglich $\underline{p}_{Ke}$ und $\underline{p}$. Für Umriß bei Betrachtung von $\underline{p}$ gilt nach (f13):

$$\underline{X}(u) = \underline{m}_{Ku} + \frac{r_{Ku}^2}{(\underline{p} - \underline{m}_{Ku})^2}(\underline{p} - \underline{m}_{Ku}) + r_{Ku}\sqrt{1 - \frac{r_{Ku}^2}{(\underline{p} - \underline{m}_{Ku})^2}}\,(\underline{a}\cos u + \underline{b}\sin u)\,. \qquad (f17)$$

Die unbekannten Größen $\underline{m}_{Ku}$, $\underline{p}$ und r_{Ku} errechnen sich aus:

$$\underline{m}_{Ku} = \underline{m}_{Ke} + (\underline{m}_{Ke} - \underline{p}_{Ke})\ \frac{r_{Ke}^2}{(\underline{m}_{Ke} - \underline{p}_{Ke})^2}\ ,$$

$$\underline{p} = \underline{p}_{Ke} + \frac{(\underline{m}_{Ke} - \underline{p}_{Ke})^2}{(\underline{p}_{Ke} - \underline{z})(\underline{m}_{Ke} - \underline{p}_{Ke})}\ (\underline{p}_{Ke} - \underline{z}) \qquad (f18)$$

und

$$r_{Ku} = r_{Ke}\sqrt{1 + \frac{r_{Ke}^2}{(\underline{m}_{Ke} - \underline{p}_{Ke})^2}}$$

Durch Schneiden von (f17) mit der Polarebene $E_{\underline{p}_{Ke}}$

$$(\underline{x} - \underline{m}_{Ke})(\underline{m}_{Ke} - \underline{p}_{Ke}) = 0$$

folgt eine Bestimmungsgleichung für w_i und damit für

$$\underline{k}_i = \underline{x}(w_i)\ ,\ (i = 1,2).$$

In der Gleichung

$$(\underline{m}_{Ku} - \underline{m}_{Ke})(\underline{m}_{Ke} - \underline{p}_{Ke}) + \frac{r_{Ku}^2}{(\underline{p} - \underline{m}_{Ku})^2}(\underline{p} - \underline{m}_{Ku})(\underline{m}_{Ke} - \underline{p}_{Ke}) + r_{Ku}\sqrt{1 - \frac{r_{Ku}^2}{(\underline{p} - \underline{m}_{Ku})^2}}\ \cdot$$

$$(\underline{m}_{Ke} - \underline{p}_{Ke})(\underline{a}\cos u + \underline{b}\sin u) = 0$$

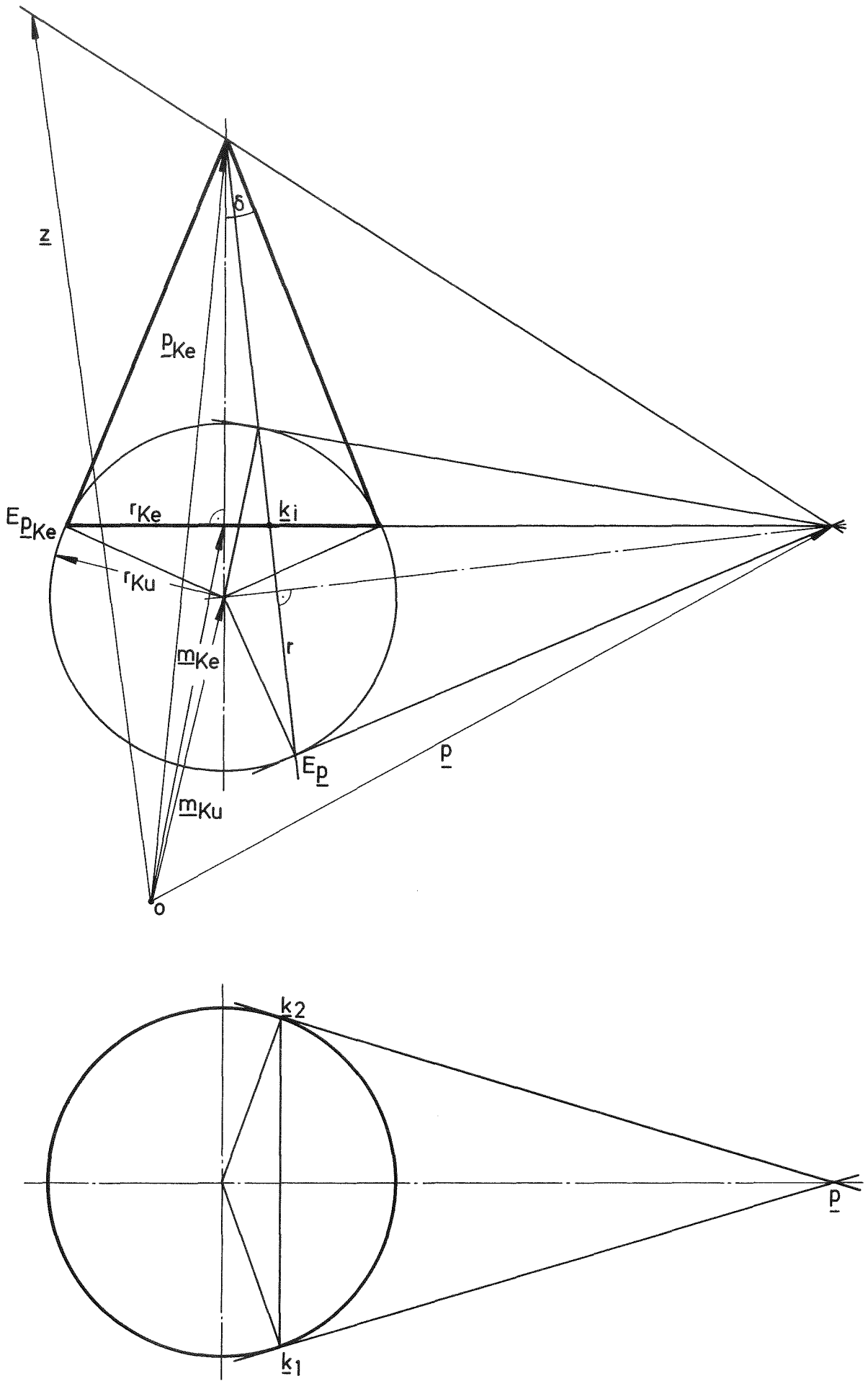


Abb. F.5: Bestimmung des wahren Kegelumrisses

werden folgende Abkürzungen eingeführt:

$$C = (\underline{m}_{Ku} - \underline{m}_{Ke})(\underline{m}_{Ke} - \underline{p}_{Ke}) \frac{r_{Ku}^2}{(\underline{p} - \underline{m}_{Ku})^2} (\underline{p} - \underline{m}_{Ku})(\underline{m}_{Ke} - \underline{p}_{Ke}),$$

$$A = r_{Ku} \sqrt{1 - \frac{r_{Ku}^2}{(\underline{p} - \underline{m}_{Ku})^2}} \; (\underline{m}_{Ke} - \underline{p}_{Ke})\, \underline{a}$$

und

$$B = r_{Ku} \sqrt{1 - \frac{r_{Ku}^2}{(\underline{p} - \underline{m}_{Ku})^2}} \; (\underline{m}_{Ke} - \underline{p}_{Ke})\, \underline{b},$$

so daß gilt

$$(A^2 + B^2) \sin^2 u + 2BC \sin u - (A^2 - C^2) = 0$$

und daraus

$$w_i = \arcsin\left(\frac{-BC \pm A\sqrt{A^2 + B^2 - C^2}}{A^2 + B^2}\right) \qquad (i = 1,2)$$

$$\underline{x}_i(u) = \underline{p}_{Ke} + u\left[(\underline{m}_{Ku} - \underline{p}_{Ke}) + \frac{r_{Ku}^2}{(\underline{p} - \underline{m}_{Ku})^2} + r_{Ku}\sqrt{1 - \frac{r_{Ku}^2}{(\underline{p} - \underline{m}_{Ku})^2}}\,(\underline{a}\cos w_i + \underline{b}\sin w_i)\right], \qquad (f19)$$

$$(i = 1,2) \quad , \quad (\underline{a}, \underline{b} \perp (\underline{p} - \underline{m}_{Ku})).$$

Bei Parallelprojektion tritt in der Bestimmungsgleichung (f18) für $\underline{p}$ an die Stelle des speziellen Projektionsstrahles $(\underline{p}_{Ke} - \underline{z})$ nur der Vektor $\underline{z}$ als bestimmende Projektionsrichtung, so daß gilt:

$$\underline{p} = \underline{p}_{Ke} + \frac{(\underline{m}_{Ke} - \underline{p}_{Ke})^2}{(\underline{m}_{Ke} - \underline{p}_{Ke})\,\underline{z}}\; \underline{z} \qquad (f20)$$

Beide Bestimmungsgleichungen (f18) und (f20) können nicht für den Fall Anwendung finden, wenn der Projektionsstrahl senkrecht auf $(\underline{m}_{Ke} - \underline{p}_{Ke})$ steht, d.h. in der Verschwindungsebene zur Polarebene $E_{\underline{p}_{Ke}}$ liegt. Da bei dieser Betrachtung aber die beiden Umrißlinien und die Kegelachse eine Ebene bilden, gilt

$$\underline{x}_i(u) = \underline{p}_{Ke} + u\left[(\underline{m}_{Ke} - \underline{p}_{Ke}) + r_{Ke}(\underline{a}\cos w_i + \underline{b}\sin w_i)\right],\ (i=1,2), \quad \text{(f21)}$$

$$(\underline{a}\ \underline{b} \perp \underline{z} \mid (\underline{p}_{Ke} - \underline{z}))$$

und

$$w_i = -\arctan\left(\frac{\underline{a}(\underline{m}_{Ke} - \underline{p}_{Ke})}{\underline{b}(\underline{m}_{Ke} - \underline{p}_{Ke})}\right) + (i-1)\pi\ ,\ (i=1,2).$$

F 3. Durchstoßpunkte einer Geraden durch die Flächen

F 3.1 Gerade - Ebene

Eine nicht zur Ebene parallele Gerade

$$\underline{x} = \underline{p}_G + u\, \underline{n}_G$$

durchstößt diese in

$$\underline{X} = \underline{p}_G - \frac{(\underline{p}_G - \underline{p}_E)\,\underline{n}_E}{\underline{n}_G\,\underline{n}_E}\,\underline{n}_G$$

F3.2 Gerade - Kugel

Für die Schnittpunkte zwischen Gerade und Kugel ergibt sich

$$\underline{x}_{1,2} = \underline{p}_G + \underline{n}_G\left[\underline{n}_G(\underline{m}_{Ku} - \underline{p}_G) \pm \sqrt{LM - r_{Ku}^2}\right]$$

mit

$$LM = (\underline{n}_G(\underline{p}_G - \underline{m}_{Ku}))^2 - \underline{n}_G^2\,(\underline{p}_G - \underline{m}_{Ku})^2 = (\underline{n}_G \times (\underline{p}_G - \underline{m}_{Ku}))((\underline{p}_G - \underline{m}_{Ku}) \times \underline{n}_G)$$

F3.3 Gerade - Zylinder

Eine Gerade und ein Zylinder haben folgende Punkte gemeinsam:

$$\underline{X}_{1,2} = \underline{p}_G + \frac{\underline{n}_G}{LN}\left[LMN - LPN \pm \sqrt{(LPN - LMN)^2 - LN(LP + LM - r_z^2)}\right]$$

wobei

$$LN = \underline{n}_G^2\,\underline{n}_z^2 - (\underline{n}_G\,\underline{n}_z)^2 = (\underline{n}_G \times \underline{n}_z)^2,$$

$$LPN = (\underline{p}_G\,\underline{n}_G)\,\underline{n}_z^2 - (\underline{p}_G\,\underline{n}_z)(\underline{n}_G\,\underline{n}_z) = (\underline{p}_G \times \underline{n}_z)(\underline{n}_G \times \underline{n}_z),$$

$$LMN = (\underline{m}_z\,\underline{n}_G)\,\underline{n}_z^2 - (\underline{m}_z\,\underline{n}_z)(\underline{n}_G\,\underline{n}_z) = (\underline{m}_z \times \underline{n}_z)(\underline{n}_G \times \underline{n}_z),$$

$$LP = \underline{p}_G^2\,\underline{n}_G^2 - (\underline{p}_G\,\underline{n}_z)^2 = (\underline{p}_G \times \underline{n}_z)^2$$

und

$$LM = \underline{m}_i^2 \underline{n}_i^2 - (\underline{m}_i \underline{n}_i)^2 = (\underline{m}_i \times \underline{n}_i)^2$$

F3.4 Gerade - Kegel

Gerade und Kegel schneiden sich in

$$\underline{X}_{1,2} = \underline{p}_G + \frac{\underline{n}_G}{LN}\left[LPK - LPG \pm \sqrt{(LPG - LPK)^2 - LN \cdot LPP}\right]$$

mit

$$LN = (\underline{n}_G(\underline{m}_{Ke} - \underline{p}_{Ke}))^2 - (\underline{m}_{Ke} - \underline{p}_{Ke})^2 \underline{n}_G^2 \cos^2\delta,$$

$$LPG = ((\underline{m}_{Ke} - \underline{p}_{Ke})\underline{p}_G)(\underline{n}_G(\underline{m}_{Ke} - \underline{p}_{Ke})) - (\underline{m}_{Ke} - \underline{p}_{Ke})^2(\underline{p}_G \underline{n}_G)\cos^2\delta,$$

$$LPK = ((\underline{m}_{Ke} - \underline{p}_{Ke})\underline{p}_{Ke})(\underline{n}_G(\underline{m}_{Ke} - \underline{p}_{Ke})) - (\underline{m}_{Ke} - \underline{p}_{Ke})^2(\underline{p}_{Ke}\underline{n}_G)\cos^2\delta$$

und

$$LPP = ((\underline{p}_G - \underline{p}_{Ke})(\underline{m}_{Ke} - \underline{p}_{Ke}))^2 - (\underline{m}_{Ke} - \underline{p}_{Ke})^2(\underline{p}_G - \underline{p}_{Ke})^2 \cos^2\delta .$$

F4. Flächennormalen

Die Normalen zu den hier untersuchten Flächen sind gegeben durch den Gradienten im Punkt $\underline{x}$:

Ebene:

$$\text{grad } Q_E(\underline{x}_E) = \underline{n}_E$$

Kugel:

$$\text{grad } Q_{Ku}(\underline{x}_{Ku}) = (\underline{x}_{Ku} - \underline{m}_{Ku})$$

Zylinder:

$$\text{grad } Q_Z(\underline{x}_Z) = (\underline{x}_Z - \underline{m}_Z) - ((\underline{x}_Z - \underline{m}_Z)\underline{n}_Z)\underline{n}_Z$$

Kegel:

$$\text{grad } Q_{Ke}(\underline{x}_{Ke}) = \frac{(\underline{x}_{Ke} - \underline{p}_{Ke})}{|\underline{x}_{Ke} - \underline{p}_{Ke}|} - \frac{(\underline{m}_{Ke} - \underline{p}_{Ke})}{|\underline{m}_{Ke} - \underline{p}_{Ke}|} \frac{1}{\cos\delta}$$

Literatur

/ 1/ APPEL, A., STEIN, A.: A System for the Interactive Design of Polyhedra, Proc. Online 72 2 (1972), S. 383

/ 2/ ADAMS, J.A.: Geometric Concepts for Computer Graphics, NTIS,AD-750743 (1972) Sept.

/ 3/ ARGYRIS, J.H., GRIEGER, I.: Die Anwendung von Interaktiver Computer-Graphik bei der Berechnung von Tragwerken, Angewandte Informatik (1972) 4, S. 173

/ 4/ ANDERSON, R.H., FORBER, D.J.: Extensions to the PL/1 Language for Interactive Computer Graphics, RM - 6028 - ARPA (Rand Corp., Santa Monica) (1970) Jan.

/ 5/ AHUJA, D.V., COONS, S.A.: Interactive Graphics in Data Processing Geometry for Construction and Display, IBM Systems Journal 7 (1968) 3, S. 188

/ 6/ ALCOCK, SHEARING and Partners: GENESYS Reference Manual, The GENESYS Centre, Loughborough (1973)

/ 7/ BOUKNIGHT, J., KELLEY, K.: Algorithm for Producing Half-Tone Computer Graphics, Presentations with Shadows and Movable Light Sources, AFIPS 36 (1970), S.1

/ 8/ BOEHM,W.: Darstellung von Flächen in der Datenverarbeitung, Angewandte Informatik (1971) 8 , S. 373

/ 9/ BOEHM, W.: Datenstrukturen für die Erfassung von Flächen, Angewandte Informatik (1971) 9, S.419

/ 10/ BAATZ, U.: Bildschirmunterstütztes Konstruieren: Funktionsfindung, Prinziperarbeitung, Gestaltung und Detaillierung mit Hilfe graphischer Datenverarbeitungsanlagen, Diss. TH Aachen (1971) Mai

/ 11/ BEZIER, P.E.: Example of an Existing System in the Motor Industry: The UNISURF System, Proc. Royal Society, London 321 (1971), S.207

/ 12/ BJAALAND, H.K., NELSON, M.F.: ICES TOPOLOGY 1 Engineering Users Manual, MIT: R72 - 65 (1972) Okt.

/ 13/ BEITZ, W., SCHNELLE, E.: Rechnerunterstützte Informationsbereitstellung für den Konstrukteur, Konstruktion 26 (1974) 2, S.46

/ 14/ BAUM, M.: Bildschirmunterstützte Informationshandhabung bei automatisierten Planungssystemen, Diss. RWTH Aachen (1972) Jan.

/ 15/ BALOGH, L.: Ein Beschreibungssystem für rotatiossymmetrische Werkstücke unter besonderer Berücksichtigung des Rechnereinsatzes in Konstruktion und Fertigungsplanung, Diss. TU Berlin (1969)

/ 16/ BOULLIER, P., GROS, J., JANCENE, P., LEMAIRE, A., PRUSKER, F., SALTEL, E.: METAVISU-A General Purpose Graphic System, Nake (ED.), Graphic Languages, North-Holland, Proc. IFIP Working Conf. (1972), S.244

/ 17/ BUTZ, H.-W.: Entwurfs- und Detailzeichnungserstellung mit Hilfe graphischer Datenverarbeitungsanlagen, Diss. RWTH Aachen (1974) Sept.

/ 18/ BURAU, W.: Algebraische Kurven und Flächen (Bd. 2): Algebraische Flächen 3. Grades und Raumkurven 3. und 4. Grades, Sammlung Goeschen, Bd. 436/436A, Verlag W. de Gruyter u. Co. (1962)

/ 19/ BOUMANS, S., LAUER, A.: Movi - 3D: Ein Plottprogramm zur 3-dimensionalen Darstellung kinetischer Vorgänge in 16 mm-Filmen, Juel - 1029 - RG (1973) Dez.

/ 20/ BECKER, E.: Ein Beitrag zur interaktiven Verarbeitung analytisch beschreibbarer Flächen mittels Rechner-Graphik, Diss. RWTH Aachen (1975) Juli

/ 21/ BOLLIET, L.: Compiler Writing Techniques, aus Programming Languages, F. Genuys (ED.) (1968), S. 113

/ 22/ BRAID, I.C.: The Synthesis of Solids Bounded by Many Faces, CACM 18 (1975) 4, S. 209

/ 23/ BRANKAMP, K., WIENDAHL, H.P.: Rationalisierungsmöglichkeiten in Konstruktionsbereich, VDI-Berichte Nr. 152 (1970), S. 5

/ 24/ BARNHILL, R.E., RIESENFELD, R.F.: Computer Aided Geometric Design, Academic Press New York (1974)

/ 25/ BECKER, J., WOIT, J., ZAJONC, H.: Einsatz der elektronischen Datenverarbeitung in der Bundesrepublik 1970, KFK-Ext. 2/71-2 (1971) Dez.

/ 26/ COONS, S.A.: Surfaces for Computer-Aided Design of Space Forms, MIT - MAC - TR - 41 (1967) June

/ 27/ COONS, S.A.: An Outline of the Requirements for a CAD System, Proc. SJCC (1963), S. 299

/ 28/ COMBA, P.G.: A Language for Three-Dimensional Geometry, IBM Systems Journal 7 (1968) 3, S. 292

/ 29/ COMBA, P.G.: A Procedure for Detecting Intersections of Three-Dimensional Objects, JACM 15 (1968) 3, S.354

/ 30/ CLEBSCH, A.: Vorlesungen über Geometrie, B.G. Teubner Verlag, Leipzig (1891)

/ 31/ CAYLEY, A.: On the Developable Surfaces which Arise from Two Surfaces of the Second Order, The Cambridge and Dublin Mathematical Journal, Cambridge, MacMillan u. Co 5 (1850), S. 46

/ 32/ CORDES, R.M., BREWER, J.A.: A User's Guide to the GETAM-II, Interactive Graphics System, Louisiana State University, Baton Rouge (1974)

/ 33/ CALCOMP GmbH: Programme für CALCOMP-Plotter der Serien 500, 600 und 700

/ 34/ DODD, G.G.: APL-A Language for Associative Data Handling in PL/1, Proc. FJCC (1966), S. 677

/ 35/ DEMIC, J.W. Perspective Geometry and Computer Graphics: Techniques and Applications in Computer-Aided Design, Intern. Computing Symposium, Davos (1973) Okt., S. 435

/ 36/ DANIELS, R.L., HALL, E.J.: ICES Project-General Description, Mit R 68 - 19 (1968) März

/ 37/ ENGELI, M.E.: A Language for 3D Graphics Applications, Intern. Computing Symposium, Davos (1973) Okt., S. 460

/ 38/ ENCARNACAO, J.L.: PRADIS - ein Programmsystem für räumliche Darstellung auf Displays, Elektronische Datenverarbeitung (1970) 7, S. 315

/ 39/ ENCARNACAO, J.L.: A Survey of a New Solution for the Hiddenline Problem, Symposium Interactive Computer Graphics Delft (1970), Okt.

/ 40/ ENCARNACAO, J.L.: Untersuchungen zum Problem der rechnergesteuerten räumlichen Darstellungen auf ebenen Bildschirmen, Diss. TU Berlin, (1972) Juli

/ 41/ ENCARNACAO, J.L.,
WALDSCHMIDT, K.,
GILOI, W.,
SANITER, J.,
STRASSER, W.: Programmierungs- und gerätetechnische Realisierung einer 4 x 4-Matrix für Koordinatentransformationen auf Computer-Bildschirmgeräten, Elektron. Rechenanlagen 14 (1972) 5, S. 206

/ 42/ ENCARNACAO, J.L.: Computer Graphics, Oldenbourg Verlag, München (1975)

/ 43/ EVANS, D.C.: Graphical Man/Machine Communications, Utah University, AD - 754 282, (1972) Juni

/ 44/ EKLINS, D.R.,
WOLF, E.J.: GRAFTRAN: Graphic Extensions to FORTRAN, Naval Postgraduate School Monterey, Calif., AD - 758 681 (1972) Dez.

/ 45/ ENDERLE, G.: Definition, Übersetzung und Anwendung benutzerorientierter Sprachen als Erweiterung von PL/1 in dem System für das rechnerunterstützte Entwickeln und Konstruieren REGENT, KFK 2204 (1975), Sept.

/ 46/ ENDERLE, G.
LEINEMANN, K.
SCHLECHTENDAHL, E.G.,
SCHNAUDER, H.,
SCHUMANN, U.,
SCHUSTER, R.: Fähigkeiten und Implementierung der GRAPHIC-Sprache (ACM-Tag. Erlangen 12/72) KFK-EXT. 8/72 - 7 (1972) Nov.

/ 47/ ENDERLE, G.
SCHLECHTENDAHL, E.G.,
SCHUMANN, U.,
SCHUSTER, R.: Design Principles of the GRAPHIC System, KFK 1722 (1972) Dez.

/ 48/ ENDERLE, G.,
SCHLECHTENDAHL, E.G.:
The Design of the integrated CAD System REGENT, Proc. of the Workshop on CAD Systems, Toulouse (France) (1974) Dez.

/ 49/ Engineers Guide to Ices COGO I, First Edition R 67 - 46, MIT (1967)

/50/ FORREST, A.R.: Current Developments in the Design and Produktion of 3D Curved-Objects-Computational Geometry, Proc. Royal Society, London 321 (1971), S. 187

/51/ FURUKAWA, K.: A Comparison of Data Structures in Computer Graphics, Information Processing in Japan 11 (1971), S.44

/52/ FURUKAWA, S.: A Solution to Interference Problem of Polyhedra, Information Processing in Japan, Faculty of Engineering, Yamanashi University, T 14,(1973) 3, S.116 - 123

/53/ FALKOFF, A.D., IVERSON, K.E.: The Design of APL, IBM Journal of Research and Development (1973) Juli S. 324

/54/ GOURAUD, H.: Continuous Shading of Curved Surfaces, IEEE Transaction on Computers 20 (1971) 6, S.623

/55/ GRAY, J.C.: Compound Data Structure for Computer Aided Design-A Survey, Proc. A. C. M. National Meeting 1967 (1967) S. 355

/56/ GRAY, J.C., LANG, C.A.: ASP-A Ring Implemented Associative Structure Package, CACM 11 (1968) 8, S.550

/57/ GLATZER, V., ENCARNACAO,J.: DATAS-Datenstrukturen in Assoziativer Speicherung, Angewandte Informatik(1972) 9, S.418

/58/ GILOI, W.K., ENCARNACAO, J., KESTNER, W.: APL-G APL Extended for GRAPHICS, Online 72 Conference Proc.(1972), S.579

/59/ GALIMBERTI, R., MONTANARI,U.: An Algorithm for Hidden Line Elimination, CACM 12(1969) Apr., S.206

/60/ GRABOWSKI,H., BUTZ,H.-W.: Rechnerunterstütztes Entwerfen von Produkten Industrie-Anzeiger 96(1974) 34, S.755

/61/ GROTEMEYER, K.P.:Analytische Geometrie, Sammlung Goeschen Bd.65/65a, Verlag W.de Gruyter u.Co., (1964)

/62/ GRABOWSKI, H., WESSEL, H.-J.: Rechnerunterstützte Variantenkonstruktion von Baugruppen. Konstruktion 27(1975), S.265-269

/63/ GRIMM, W.: Vorlesung über Geometrie für computerunterstütztes Konstruieren. UNI Karlsruhe (1974)

/64/ GRILL,H.: Informationssystem Technik (IST) Vorentwurf der Grafik-Verwaltung, TU Berlin, Inst.für Allg.Bauingenieurmethoden (1975) Febr.

/ 65/ GINO-F, Computer-Aided Design Centre, Cambridge, CADC/ICL/73/02 (1973)

/ 66/ HAWKINSON, B.: A Computer-Controlled Microfilm System, Datamation (1970) März, S. 119

/ 67/ HERZOG, B.: A Computer Graphics Language (Drawl 70), AD 715 952 (1970) Aug.

/ 68/ HERZOG, B.: Special Issue on Computer Graphics, Proc. of the IEEE 62 (1974) 4

/ 69/ HURWITZ, A., CITRON, J.P., YEATON, J.B.: GRAF: Graphic Additions to Fortran, Proc. SJCC (1967), S. 533

/ 70/ HERBERTZ, R.: Datenverarbeitung in der Konstruktion 1973; Konstruktion 25 (1973) 10, S. 406

/ 71/ HEROLD, W.-D.: Der Einsatz der graphischen Datenverarb. in den Konstruktionsphasen des Entwerfens und Detaillierens, Diss. Uni Bochum (1974)

/ 72/ ISAKOWER, R.I.: Status of Interactive Graphics, Hardware and Software, NTIS, AD-784 067 (1974) Juni

/ 73/ IBM CORP.: OS PL/1 Checkout-Optimizing Compilers Language Reference Manual, SC33 - 0009 - 2

/ 74/ IBM CORP.: PLAN-Problem Language Analyser, Users Introduction, Application Description Manual, IBM Form: H20 - 0626, H20 - 0490

/ 75/ JONES, C.B.: A New Approach to the Hidden Line Problem, The Computer Journal 14 (1970) 3, S. 232

/ 76/ JORRAND, P.: On some Basic Concepts for Extensible Programming Languages, IBM France Scientific Centre Grenoble (1971)

/ 77/ JEGER, M., BECKMANN, B.: Einf. in die vektorielle Geometrie und Lineare Algebra, Birkhäuser Verlag (1967)

/ 78/ KIRA, K.: Detection of Outlines and their Use in Hiddenline Elimination, Information Processing in Japan 14 (1974) 3, S. 124 - 130

/ 79/ KULSRUD, H.E.: A General Purpose Graphic Language, CACM 11 (1968) 4, S. 247

/ 80/ KURTH, J.: Rechnerorientierte Werkstückbeschreibung
Diss. TU Berlin (1971)

/ 81/ KUBERTH, B.,
SZABO, J.,
GIULIERI, S.: The Perspective Representation of Functions of Two Variables,
JACM 15 (1968) 2, S. 193 - 204

/ 82/ KRAUS, H.,
WIESMÜLLER, S.: Sicograph, Ausgabeterminal für flächenhafte Farbdarstellungen,
Siemens Zeitschrift (1975) 1, S. 34 - 40

/ 83/ KRÜGER, G.: Vorlesung über graphische Datenverarbeitung,
UNI Karsruhe (1974)

/ 84/ KAMIUCHI, H.: About the Outlines of Three Dimensional Objects in Computer Graphic Processing,
Information Processing in Japan 10 (1970), S. 121

/ 85/ KAMIUCHI, H.: About the Three-Dimensional Hidden-Line Problem,
Information Processing in Japan 10 (1970),
S. 81

/ 86/ LACOSTE, J.-P.: Rechnerangepaßte Werkstückdarstellung für den automatisierten Produktionsprozeß,
Diss. TH Aachen (1972)

/ 87/ LÜPERTZ, H.: Rationalisierung des Konstruktionsprozesses durch Anwendung neuer zeichnerischer Darstellungsarten,
Konstruktion 23 (1971) 12, S. 462

/ 88/ LUH, J.Y.S.,
KROLAK, R.J.: A Mathematical Model of Mechanical Part Description,
CACM 8 (1965) 2, S. 125

/ 89/ LYNCH, J.P.,
ROLAND, R.D.: Computer Animation of a Bicycle Simulation,
Proc. FJCC (1972), S. 161

/ 90/ LOUTREL, PH.: A Solution to the Hidden-Line Problem for Computer-drawn Polyhedra,
New York University, Ph. D. Thesis, 1968, 69 - 583 (1967) Sept.

/ 91/ LEINEMANN, K.: Ein Ansatz zur formalen Objektbeschreibung für das rechnergestützte Entwickeln und Konstruieren,
Angewandte Informatik (1974) 8, S. 403 - 406

/ 92/ LEINEMANN, K.,
SCHUMANN, U.: Ein Beitrag zu Grundlagen des rechnergestützten Entwurfs und einer Konstruktionssprache,
KFK-EXT. 8/72 - 5 (1973) Febr.

/ 93/ LEINEMANN, K., SCHUMANN, U.: Kospra-Entwurf einer Konstruktionssprache zur Beschreibung der Geometrie technischer Objekte, KFK-EXT. 8/72 - 6 (1973) Febr.

/ 94/ LOPEZ, L.A.: POLO-Problem Oriented Language Organizer, Computers and Structures 2 (1972), S.555

/ 95/ MATSUSHITA, Y.: Hidden Lines Elimination for a Rotating Object, CACM 15 (1972) April, S. 245

/ 96/ MC. GRATH, F.J.: A Method for Eliminating Hidden Lines with Polyhedra, Simulation 16 (1971) Jan., S. 37

/ 97/ MAC CALLUM, K.J.: Surfaces for Interactive Graphical Design, The Computer Journal 13 (1970) Nov., S. 352

/ 98/ METZGER, W.: Gesetze des Sehens, Verlag: Kramer,Frankfurt/M. (1936)

/ 99/ MAXWELL, E.A.: General Homogeneous Coordinates in Space of Three Dimensions, Verlag University Press, Cambridge (1959)

/100/ MACKWORTH, A.K.: Interpreting Pictures of Polyhedral Scenes, Artificial Intelligence 4 (1973), S. 121

/101/ MESHKOWSKI, H., AHRENS,I.: Theorie der Punktmengen, Bibliographisches Institut, Mannheim (1974)

/102/ MOUNTFORD, D.H., SHAW, L., ABRAHAMS, G., MAYOR, P.: EUKLID, Conference on Computer Aided drafting Systems at St. Johns College, Cambridge (1973) April

/103/ MBB: Handbuch 2D und 3D GEOLAN, Messerschmidt - Boelkow - Blohm (1975)

/104/ MÜLLER, E., KRUPPA, E.: Lehrbuch der darstellenden Geometrie, Springer Verlag, Wien (1961)

/105/ NEWMAN, W.M.: Display Procedures, CACM 14 (1971) 10, S. 651

/106/ NEWMAN, W.M., SPROULL, R.F.: Principles of Interactive Computer Graphics, Mc. Graw-Hill Book Company (1973)

/107/ NEWMAN, W.M.: An Informal Graphics System on the LOGO Language, National Computer Conference (1973), S. 651 - 655

/108/ NEISS, F.: Analytische Geometrie, Springer Verlag (1950)

/109/ NAKAMAE, E., NISHITA, T.: An Algorithm for Hidden Line Elimination of Polyhedra, Information Processing in Japan 12 (1972), S. 134

/110/ NAKE, F., PIEKE, A.: Entwurf und Implementierung einer grafischen Programmiersprache, ACM-German Chapter Lectures III - 1972, S. 103

/111/ NOTLEY, M.G.: A Graphical Picture Drawing Language, The Computer Bulletin (1971), S. 68

/112/ NELSON, D.A.: Whatever happened to PL/1, The Diebold Research Program, Conf. Proc. Meeting XXIX, Genova, Doc. No. EC29 (1973)

/113/ NEUHOLD, E.J.: The Formal Description of Programming Languages, IBM Systems Journal 10 (1971) 2, S. 86

/114/ NEES, G.: Generative Computergraphik, Siemens Verlag, ISBN 3-8009-1046-2 (1969)

/115/ PESCHKE, J.V.: Erweiterung von Programmiersprachen durch ihre Benutzer, Angewandte Informatik (1971) 7, S. 331

/116/ PAPPERITZ, E.: Problem der kürzesten und weitesten Entfernung eines Punktes von einer Oberfläche 2. Ordnung, Diss. UNI Leipzig (1883)

/117/ PAHL, J.P.: ISB-Informationssystem für das Bauwesen, BMBW-FB DV 72-09 (1972) Dez.

/118/ ROSS, D.T.: The AED Approach to Generalized Computer-Aided Design, Proc. ACM National Meeting, (1967), S. 367

/119/ ROSS, D.T., RODRIGUEZ, J.E.: Theoretical Foundations for the CAD System, Proc. SJCC (1963), S. 306

/120/ ROOS, D.: ICES System Design, MIT, Dep. of Civil Engineering, the MIT Press, 2nd Ed. (1967)

/121/ ROSEN, S.: Programming Systems and Languages 1965 - 1975, CACM 15 (1972) 7, S. 591

/122/ ROBERTS, L.G.: Machine Perception of Three-Dimensional Solids, MIT-TR-315, MIT, (1963) Mai

/123/ RICCI, A.: A Constructive Geometrie for Computer Graphic, The Computer Journal 16 (1973) 2, S. 157

/124/ ROTHENBERG,R.D.: Graphische Informationshandhabung für das rechnerunterstützte Konstruieren. DISS. RWTH Aachen, 1975

/125/ REUTTER, F.,
HEYNE, E.: Untersuchungen über die automatische Lösung von Aufgaben der konstruktiven Geometrie, Forschungsberichte des Landes NRW, Nr. 2300 (1973)

/126/ SCHRADER, K.H. MESY, Systemprogrammierung mit PL/1, Angewandte Informatik (1971) 7, S. 321

/127/ SUTHERLAND, I.E.: SKETCHPAD-A Man-Machine Graphical Communication System, Proc.-SJCC (1963), S.329

/128/ SUTHERLAND, I.E.: Ten Unsolved Problems, Datamation 12 (1966) 5, S.22

/129/ SUTHERLAND, I.E.,
SPROULL, R.F.,
SCHUMACKER, R.A.: A Characterization of ten Hidden-Surface Algorithms, Computing Surveys 6 (1974) März, S. 1

/130/ SOLNTSEFF, N.: A Classification of Extensible Programming Languages, Information Processing Letters 1 (1972) Febr., S.91

/131/ SMITH, D.N.: GPL/I-A PL/1 Extension for Computer Graphics, Proc. SJCC (1971), S.511

/132/ SCHNELLE, E.: Beitrag zur Entwicklung eines Sprachsystems für das rechnerunterstützte Konstruieren, Zeitschrift für wirtsch. Fertigung 66 (1971) 7 S. 346

/133/ SOOP, K.: The Design and Use of a PL/1 Based Graphical Programming Language, Online 72 Conference Proc. (1972) S.601

/134/ SAMMET, J.E.: An Overview of Programming Languages for Specialized Application Areas, AFIPS Conf. Proc. 40 (1972), S. 299

/135/ STAUDE, O.: Analyt. Geometrie des Punktes, des Kegelschnittes und der Fläche zweiter Ordnung, Band 1 + 2, B.G. Teubner Verlag, Leipzig (1910)

/136/ STAUDE, O.: Die Raumkurven 4. Ordnung 1. Spezies, Encyklopädie der mathem. Wissenschaften mit Einschluß ihrer Anwendungen, 3. Band (1910), S. 237

/137/ SCHÜTZE, R.: Einsatzmöglichkeiten von Datenverarbeitungsanlagen zur Ausgabe graphischer Information, Elektron. Rechenanlagen 15 (1973) 6, S. 284

/138/ SCHÖRNER, E.: Darstellende Geometrie, Carl Hanser Verlag, München (1973)

/139/ SAMMET, J.E.: Programming Languages: History and Future, CACM 15 (1972) 7, S. 601

/140/ SIEMENS SYSTEM 4004:
Informationssystem Technik (IST),
Programmierhandbuch und Kommandosprache, (1973)

/141/ SIMON, R.: Rechnerunterstütztes Konstruieren:
Eine Möglichkeit zur Rationalisierung im Konstruktionsbereich,
Diss. RWTH Aachen (1968)

/142/ Space-Plot,
H.Wenz, Ingenieure und Physiker, Frankfurt/M. (1973)

/143/ SCHLECHTENDAHL, E.G.:
Comparison of Integrated Systems for CAD,
Computer Aided Design Intern. Conf. 8.-11.4.74,
IEE Conf. Publication No. 111 (1974) Apr., S.111

/144/ SCHLECHTENDAHL, E.G.,
SCHUMANN,U.,
ENDERLE, G.,
KATZ, W.,
SCHNAUDER, H.,
SCHUSTER, R.: Erfahrungen mit dem Programmsystem ICES bei ingenieurtechnischen Anwendungen,
KFK 1586 (1972)Mai

/145/ SCHLECHTENDAHL, E.G. (Ed.):
Erster - siebter REGENT-Halbjahresbericht,
KFK-Ext. (8/72-2, 8/72-4, 8/73-1, 8/73-4, 8/74-3, 8/74-5, 8/75-2) (1972-75)

/146/ SCHLECHTENDAHL, E.G.:
Programmiersprachen im CAD-Bereich,
CAD-Mitteilungen 1/1973,
GfK, Karlsruhe (1973), S.276

/147/ SCHLECHTENDAHL, E.G.,
ENDERLE, G.: What can we Learn from Practical Application of Ices for the Design of Integrated CAD System Nucleus?
Colloque international sur les systemes integres en genie civil,
Univ. de Liege 9 (1972) 8

/148/ SCHUMANN, U.: PLOTHL - ein FORTRAN-Unterprogramm zur Darstellung von Funktionen von zwei unabhängigen Variablen durch ihre Höhenlinien auf einem Plotter,
KFK 1486 (1971) Okt.

/149/ SCHUMANN, U.,
SCHUSTER, R.: MAPLIB - A Programm System for Delivery of Material Property Data to Computer Programs and Users,
International Congress: Use of Electronic Computers in Chemical Engineering, 24 (1973) Apr.

/150/ SCHUSTER, R.,
ENDERLE, G.,
LEINEMANN, K.,
SCHLECHTENDAHL, E.G.,
SCHNAUDER, H.,
SCHUMANN,U.: Sprach- und Datenstruktur des Systems GRAPHIC,
Vortrag 2. GI-Jahrestagung, Karlsruhe,2.-4.Okt. 1972
Lecture Notes in Operation Research and Math.
Systems, 78., Berlin: Springer 1973, S. 322

/151/ SCHUSTER, R.,
SCHLECHTENDAHL, E.G.,
ENDERLE, G.: GRAPHIC-Ein ICES-Subsystem zur Abbildungsmanipulation,
KFK Nachrichten 4/73 5 (1973)

/152/ SCHUSTER, R.: Darstellung der Stoffdaten des Systems MAPLIB in tabellarischer und graphischer Form,
KFK 1792 (1973) Mai

/153/ TALBOT, P.A.: Animator: An On-Line Two-Dimensional Film Animation System,
CACM 14 (1971) Apr., S. 251

/154/ TAKASAWA, Y.,
MORIGUCHI, S.,
SAKAMAKI, T.: A Graphic Manipulating Language,
Nake (Ed.), Graphic Languages,
North-Holland, Proc. IFIP Working Conf. (1972)
S. 327

/155/ THOMPSON, F.B.,
DOSTERT, B.H.: The Future of Specialized Languages,
Proc. SJCC (1972), S. 313

/156/ TJALVE E.,
ANDREASEN, M.M.: Zeichnen als Konstruktionswerkzeug,
Konstruktion 27 (1975), S. 41

/157/ VDI: Datenverarbeitung in der Konstruktion:
Methoden und Hilfsmittel; Aufgabe, Prinzip und Einsatz von Informationssystemen,
VDI-Richtlinien: VDI 2211, Blatt 1 (1973) März

/158/ VDI: Datenverarbeitung in der Konstruktion:
Methoden und Hilfsmittel; Berechnungen in der Konstruktion,
VDI-Richtlinien: VDI 2211, Blatt 2 (1973) März

/159/ VDI: Datenverarbeitung in der Konstruktion:
Methoden und Hilfsmittel; Maschinelle Herstellung von Zeichnungen,
VDI-Richtlinien: VDI 2211, Blatt 3 (1973) März

/160/ WOODSFORD, P.A.: Mathematical Methods in Computer Graphics,
Computer-Graphics Symposium Berlin (1971) Okt.

/161/ WILLIAMSON, H.: Hidden-Line Plotting Program,
CACM 15 (1972) Febr., S. 100

/162/ WILLIAMS, R.: A General Purpose Graphical Language,
Nake (Ed.), Graphic Languages,
North-Holland, Proc. IFIP Working Conf. (1972),
S.334

/163/ WILLIAMS, R.: A Survey of Data Structures for Computer Graphics Systems,
Computing Surveys 3 (1971) März,

/164/ WOON, P.: A Computer Procedure for Generating Visible-Line Drawings of Solids Bounded by Quadric Surfaces,
New York Univ., Bronx, N.Y. AFOSR-TR-71-1613
(1970) Dez.

/165/ WYLIE, C.,
ROMNEY, G.
EVANS, D.,
ERDAHL, A.: Half-Tone Perspective Drawings by Computer,
Proc.-FJCC (1967), S. 49

/166/ WEGNER, P.: The Vienna Definition Language,
Computing Surveys 4 (1972) 1, S. 5

/167/ WEISS, R.A.: BE VISION A Package of IBM 7090 FORTRAN Programs to draw Orthographic Views of Combinations of Plane and Quadric Surfaces,
Journal of the ACM 13 (1966) 2, S. 194

/168/ WARNOCK, J.E.: A Hidden Surface Algorithm for Computer Generated Halftone Pictures,
University of Utah 69 - 19.002 (1969) 6

/169/ WATKINS, G.S.: A Real Time Visible Surface Algorithm,
Utah University, AD - 762004 (1970) 6

/170/ WATKINS, S.L.: Masked Three-Dimensional Plot Program with Rotations,
CACM 17 (1974) 9, S. 520

/171/ WRIGHT, T.: Visible Surface Plotting Program,
CACM 17 (1974) 3, S. 152 - 157

/172/ ZWARG, S.M.: SAILING an Example of Computer Animation and Iconic Communication,
SJCC (1972), S. 1005

ERRATUM

Seite 95:

Hinter dem Statement

```
DO R=45. TO 75. BY 10.;
```

in Abb. 4.8 ist folgende Zeile zu ergänzen:

```
SET KUGEL=BALL(POINT(50.,50.,50.),R);
```

Seite 192:

In Gl.(f1) muß für $(\underline{n}_{E1}\ \underline{n}_{E2})\ \underline{n}_{E2}$ der Ausdruck $(\underline{n}_{E1}\ \underline{n}_{E1})\ \underline{n}_{E2}$ stehen.

Seite 193:

In Gl.(f3) fehlt r_Z vor den Klammerausdrücken zu den Winkelfunktionen.

Seite 200:

$$LA = (\underline{r}_a(\underline{p}_{Ke} - \underline{m}_{Ku}))^2 - \underline{r}_a^2(\underline{p}_{Ke} - \underline{m}_{Ku})^2$$

$$LB = (\underline{r}_b(\underline{p}_{Ke} - \underline{m}_{Ku}))^2 - \underline{r}_b^2(\underline{p}_{Ke} - \underline{m}_{Ku})^2$$