WERKZEUGE ZUR MODELLGESTEUERTEN BILDANALYSE UND WISSENSAKQUISITION*
- Das System ERNEST -

F. Kummert, H. Niemann, G. Sagerer, S. Schröder
Lehrstuhl für Informatik 5 (Mustererkennung)
FAU Erlangen-Nürnberg
Martensstraße 3, 8520 Erlangen

Abstract: Aufgrund der Hardware-Entwicklungen der letzten Jahre werden wissensbasierte Bildanalysesysteme zunehmend auch für industrielle Anwendungen interessant. Im folgenden Artikel wird ein Systemrahmen für derartige Anwendungen vorgestellt, der auch eine Wissensakquisitionskomponente auf CAD-Daten basierend umfaßt.

1. Einleitung

Das Ziel bei der Entwicklung von Bildanalysesystemen besteht darin, aus einem Bild, einem Stereo-Bildpaar oder aus zeitlichen Bildfolgen automatisch eine symbolische Beschreibung relevanter Bildinhalte zu erzeugen. Die Art dieser Beschreibung insbesondere auch die Festlegung relevanter Bildinhalte sind dabei abhängig von der konkreten Aufgabe, dem Problemkreis, zu wählen. So sind bei der Interpretation von Bildern aus dem medizinischen Bereich diagnostische Merkmale und Aussagen zu extrahieren, bei der Qualitätskontrolle im industriellen Bereich Defekte zu lokalisieren und zu beschreiben, während für die Steuerung eines Roboters eine Abfolge von Befehlen die adäquate "Bildinterpretation" ist. Um derartige Interpretationen zu gewinnen, wird bei komplexen Anwendungen "Wissen" über den Problemkreis explizit für das Analysesystem in einem Modell respektive der "Wissensbasis" des Systems gespeichert. Neben diesem Modul umfassen Bildanalysesysteme weitere Einheiten: Methoden zur Vorverarbeitung und Segmentierung der Bilder, eine Datenbank zur Aufnahme der Zwischen- und Endergebnisse, sowie einen Kontrollalgorithmus zur Steuerung des Analyseprozesses /1/. Bis auf die "Methoden" entspricht diese Aufteilung der eines sogenannten Expertensystems. Beide Systemtypen werden durch eine "Erklärungskomponente" vervollständigt, die es einem Benutzer ermöglichen soll, die Resultate eines Analyseprozesses anhand der Wissensbasis, des Analyseablaufs und der Zwischenergebnisse zu hinterfragen /2/. Diese einzelnen Module können für ein konkretes System mehr oder minder unabhängig voneinander betrachtet werden. Die Effizienz eines Systems hängt aber entscheidend von all diesen Komponenten und ihrem Zusammenspiel ab. Eine Einteilung kann man zum Beispiel anhand des in einem Systems verwendeten Wissensrepräsentationsschema treffen. Danach sind zu unterscheiden: regelbasierte, syntaktische Ansätze, relationale Datenbanken, Prädikatenlogik erster Ordnung, Frames, semantische Netzwerke. Beipiele für jeden dieser Ansätze findet man

*Die dargestellte Arbeit wird von der DFG und der Siemens AG Erlangen unterstützt

in /3,4/. Als Grundlage für den Kontrollalgorithmus werden Metaregeln, der A*-Algorithmus, ATN's /1/ oder spezielle für einen Problemkreis entwickelte Verfahren genutzt. Um Entscheidungen während eines Analyseprozesses zu bewerten bzw. um weitere Analyseschritte in ihrer Dringlichkeit zu gewichten, werden häufig a priori Fakten, z.B. gewichtete Regeln in der Wissensbasis, seltener a posteriori Fakten, also vom zu analysierenden Datenmaterial abgeleitete Bewertungen, genutzt. Als Schemata werden dazu Fuzzy-Sets, Wahrscheinlichkeiten, certainty factors oder spezielle heuristische Funktionen eingesetzt. Welches Wissensrepräsentationsschema, welcher Kontrollalgorithmus und welches Bewertungsschema gewählt wird, ist zunächst unabhängig vom Problemkreis. Somit bietet es sich für einen Systementwickler an, sogenannte Systemschalen zu verwenden. Derartige "shells" sind kommerziell erhältlich, wobei jedoch neben komfortablen Benutzerschnittstellen häufig Kontrollfunktionen und Bewertungsschemata fehlen. Das "Füllen" der Schale mit den "Inhalten" des Problemkreises bleibt vollständig dem Benutzer eines solchen Systems überlassen. Ansätze zur automatischen Wissensakquisition aus Bilddaten wurden für spezielle Bereiche z.B. in /5/ gemacht.

Das in diesem Artikel vorzustellende System ERNEST (ERlanger semantisches NEtzwerk SysTem) bietet einen Systemrahmen für Musteranalyse- und Expertensysteme. Den Kern von ERNEST bildet ein semantisches Netzwerk, bei dessen Definition die epistemologische Adäquatheit /6/ sowie die Kombination von symbolischen und numerischen Informationen und Verarbeitungseinheiten eine tragende Rolle spielen. Zusätzlich zur syntaktischen und semantischen Definition aller epistemologischen Primitiva /6/ wird mit der pragmatischen Definition genau dieser Einheiten ein neuer Weg für semantische Netzwerke beschritten. Damit wurde es möglich, Kontrollprimitive, ja sogar vollständige Kontrollalgorithmen, in die Systemschale von ERNEST zu integrieren, die unabhängig von intendierten Anwendungsgebieten sind. Die vollständige Definition des Netzwerks bzgl. ihrer Syntax, Semantik und Pragmatik schuf außerdem die Möglichkeit, Primitive zur Wissensakquisition aufgrund von CAD-Daten und/oder Beobachtungen in den Formalismus aufzunehmen. Diese Konstruktionsbeschreibungen für Modelle erlauben es, mit Hilfe eines inhaltlich allgemeinen Netzwerks Wissensbasen für bestimmte Anwendungsgebiete automatisch zu generieren.

Im folgenden wird zunächst der Netzwerkformalismus vorgestellt. Der Schwerpunkt bei den weiteren Kapiteln liegt bei den Werkzeugen, die in ERNEST zur Verfügung stehen. Diese umfassen Entwicklungswerkzeuge zum Aufbau von Netzwerken für konkrete Anwendungen, Analysewerkzeuge sowie Hilfsmittel zur automatischen Wissensakquisition. Eine Erklärungskomponente steht noch nicht zur Verfügung. Am Lehrstuhl der Autoren wird das System aktuell für drei Anwendungen genutzt. Das System zur diagnostischen Analyse von Bildfolgen des Herzens ist in /8/ allerdings in einer anderen Realisierung beschrieben. Die Realisierung des sprachverstehenden Systems EVAR /Nie85/ in der ERNEST-Umgebung wurde begonnen /11/. Mit einem Schwerpunkt auf der Wissensakquisition wird ein System zur Analyse industrieller Szenen entwickelt /12/.

2. Syntaktische, semantische und pragmatische Definition des Netzwerks

Semantische (oder assoziative) Netzwerke sind spezielle markierte gerichtete Graphen, wobei die Knoten Begriffe repräsentieren und Kanten Beziehungen zwischen diesen Begriffen. Ein grundlegendes Problem für Formalismen derartiger Netzwerke besteht in der festen Auswahl von Knoten und Kanten sowie weiterer Substrukturen und Einträgen und deren eindeutige Definition bzgl. ihrer syntaktischen Struktur und ihrer Bedeutung. Nur so kann eine Netzwerksprache entwickelt werden, die einerseits unabhängig von speziellen Anwendungen ist, und andererseits aber mächtig genug ist, unterschiedliche Anwendungen aufzunehmen. Die Diskussion über erforderliche und ausreichende epistemologische Primitive soll hier nicht geführt werden. Dazu sei auf die Literatur zu diesem Problem z.B. /7/ verwiesen. Der im folgenden vorzustellende Ansatz wurde durch Arbeiten an einem Bildanalysesystem /8/ initiiert. Neben der epistemologischen Adäquatheit wird auf einfaches Nutzen kodierten "Wissens" Wert gelegt. Stark beeinflußt ist das Netzwerk in ERNEST durch KL-ONE /9/ und PSN /10/.

2.1 Syntax und Semantik

Es werden drei Typen von Knoten unterschieden: Konzepte, modifizierte_Konzepte und Instanzen. Ein Konzept ist die Modellierung eines Begriffs, eine Instanz eine bewertete Zuordnung eines Konzepts zu einem existierenden Gegenstand oder Ereignis. Jede Instanz ist eindeutig mit einem Konzept und eindeutig mit einem Gegenstand assoziiert. Üblicherweise werden während eines Analyseprozesses einem Konzept mehrere Instanzen zugeordnet, aber auch einem Gegenstand mehrere Instanzen. Modifizierte Konzepte können während einer Analyse erzeugt werden. Sie sind von ihrer Charakteristik her Konzepte, die jedoch aufgrund von Instanzen eingeschränkt sind. Sie sind somit als datenabhängige bzw. datenadaptierte Konzepte zu betrachten.
Der syntaktische Aufbau eines Konzepts ist in Bild 1 dargestellt. Er gilt auch für Instanzen und modifierte Konzepte, wobei jedoch die Listen von Defaultwerten durch eindeutige Werte ersetzt oder leer sind und Prozedurnamen durch Ergebnisse ersetzt sein müssen (bei Instanzen) bzw. können (bei modifizierten Konzepten). Im folgenden werden die wichtigsten Komponenten, d.h. Substrukturen und Einträge, beschrieben.
Jedes Konzept wird zunächst durch seine Beziehungen zu anderen Konzepten beschrieben. Dabei werden vier Kantentypen unterschieden, die jeweils mit ihrer Inversen definiert sind. Entlang der Generalisierungs-/Spezialisierungshierarchie wird analog zu KL-ONE eine Vererbung definiert. Gefordert wird jedoch, daß jedes Konzept höchstens von einem anderen Konzept **Spezialisierung** sein darf. Ein Begriff, der durch ein Konzept modelliert werden soll, kann durch die Kante **Bestandteil** in Teile zerlegt werden. Als Teile in diesem Sinn, ebenso als **Spezialisierungen**, sind aber nur Begriffe zugelassen, die auf der gleichen Abstraktionsebene liegen. So sind als Bestandteile von Bewegungsbegriffen nur Bewegungsbegriffe zulässig, von Objekten nur Objekte, aber kein Objekt kann Bestandteil einer Bewegung sein. Vielmehr konkretisiert ein Objekt eine Bewegung, wenn man davon ausgeht, daß Bewegungen über

Objektfolgen beschrieben werden. Beide Kanten, **Bestandteil** und **Konkretisierung**, werden durch die Substruktur **Kantenbeschreibung** festgelegt. Bei Bestandteilen wird zusätzlich festgehalten, ob ein Begriff nur aus einem Kontext heraus interpretierbar ist oder nicht. In einem Konzept, das einen solchen Begriff modelliert, sind in dem Eintrag **Kontext** die Konzepte einzutragen, welche den Kontext festlegen. Ist ein Bestanteil kontextabhängig, so wird dies in dem entsprechenden Eintrag der zugehörigen Substruktur vermerkt. Die Kanten **Modell** und **Modell_von** gelten für Konzepte die durch Wissensakquisition auseinander hervorgegegangen sind. Neben diesen symbolischen Beschreibungen werden Konzepte durch quantitative Eigenschaften beschrieben. Diese **Attributbeschreibungs-Substrukturen** besitzen Funktionen, die eine Werteberechnung durchführen können. Quantitative Beziehungen zwischen Bestandteilen und/oder Attributen werden durch **Strukturrelationen** definiert, die somit einen Test, den **Relationstest** als Eintrag besitzen. Ebenfalls mit der Struktur Attributbeschreibung werden **lokale_Attribute** und **Analyseparameter** festgelegt. Wie der Name besagt, sind lokale Attribute Eigenschaften eines Konzepts, die nur es selbst betreffen. Sie sind somit von der Vererbung ausgeschlossen. Analyseparameter sind Hilfsgrößen, die nur dann erforderlich sind, wenn ein Modell zu Analysezwecken verwendet wird. Sie werden bei der Analyse wie Attribute behandelt, tragen aber nicht zur intensionalen Beschreibung eines Konzepts bei. Analog dazu sind **Analyserelationen** als Ergänzung zu den Strukturrelationen zu interpretieren. Sie stellen wichtige Teilergebnisse in Bezug auf die Bewertung einer Instanz dar. Die Bewertung selbst wird in einem Konzept als Prozedur referiert. Jede Kante- und Attribut-Substruktur eines Konzepts wird eindeutig durch seine **Rolle** in einem Konzept gekennzeichnet und durch den **Definitionsbereich**, einschränkende **Selektionen** und eine **Dimension** charakterisiert. Wird eine solche Struktur nicht unverändert von der Generalisierung übernommen, so ist er als **modifiziert** zu kennzeichnen. Des weiteren enthält jede Substruktur Einträge, die der Wissensakquisition dienen (in Bild 1 schraffiert).

Bestandteile und/oder Konkretisierungen lassen sich entsprechend der Aussage eines Konzepts in **Modalitätsmengen** zusammenfassen. Jede derartige Menge charakterisiert eine zulässige Konstellation von Kanten, die als obligatorisch bzw. optional für die Aussage des Konzepts zu betrachten sind. Ergänzt wird eine Modalitätsmenge durch inhärente Teile, die für die Analyse von geringer Bedeutung aber für Reaktionen eines Systems erforderlich sind, z.B. die Angriffspunkte eines Teils für einen Roboter. Zeitliche respektive örtliche Nachbarschaften können hier kompakt in Form einer **Adjazenzmatrix** angegeben werden. Diese Beschreibung kann von Attributberechnungen und Relationstests verwendet werden, falls in deren Beschreibung der entsprechende Eintrag auf JA gesetzt ist.

Abschließend werden noch kurz die **Identifikationen** für ein Konzept behandelt. Dabei handelt es sich um Mengen von Pfaden im Netzwerk, die von einem Konzept ausgehend, ein gemeinsames Konzept als Ziel haben. Durch diese Identifikationspfade wird festgelegt, daß diese Pfade auch für Instanzen identische Objekte miteinander verbinden.

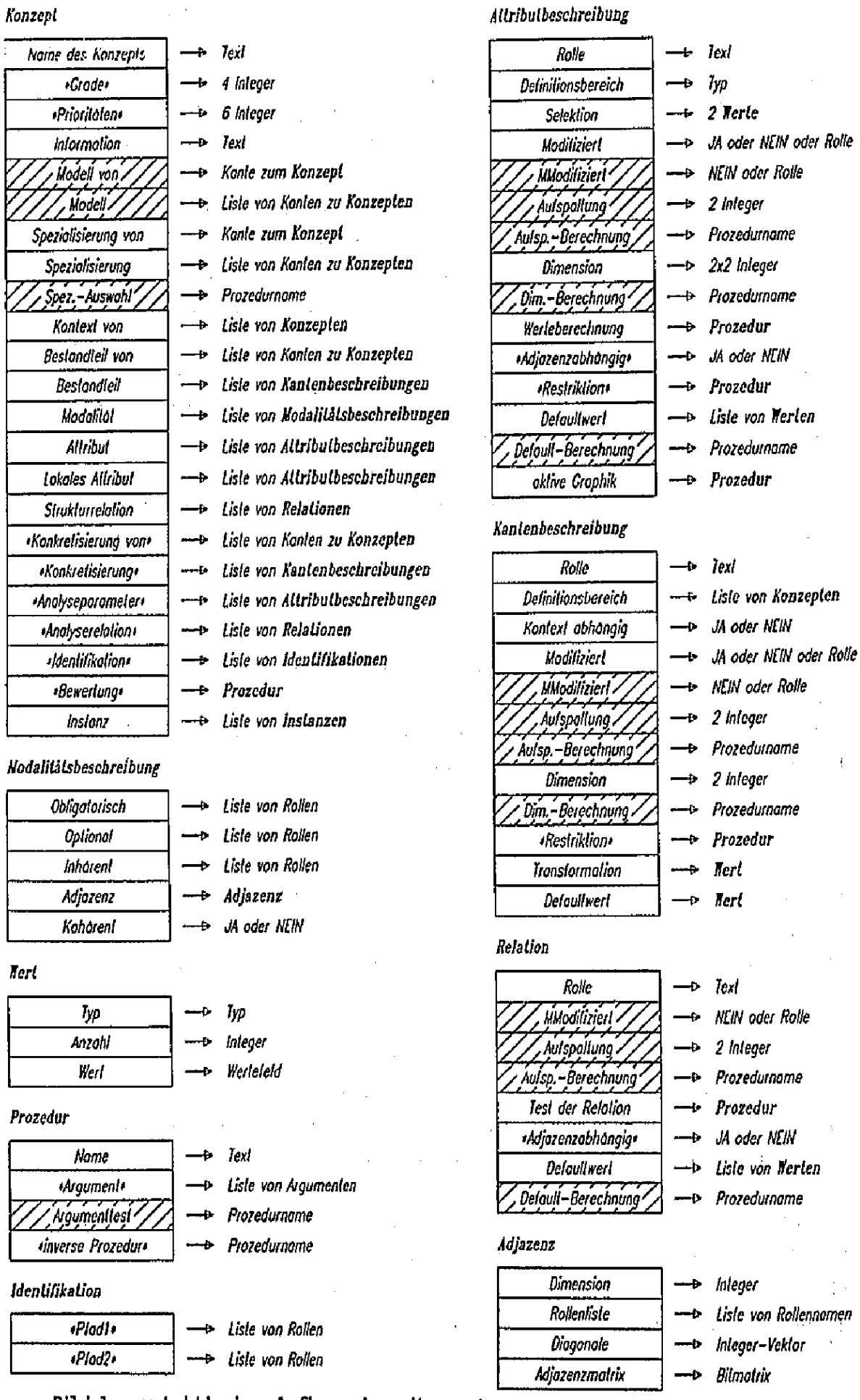

Bild 1: syntaktischer Aufbau eines Konzepts

2.2 Wissensnutzung, Konsistenz

Die Kernidee, die Fakten zu nutzen und Algorithmen zu aktivieren, die in einem Netzwerk der beschriebenen Form abgelegt sind, bilden die Produktionen zur Instantiierung. Diese Produktionen vervollständigen die Definition des Netzwerks durch seinen pragmatischen Aspekt. Im Gegensatz zu PSN sind diese Regeln zum Aufbau einer Instanz global für das Netzwerk definiert und nicht separat für jedes Konzept.

IF für ein Konzept A, Instanzen zu den Konzepten vorliegen, die in A wie folgt referiert werden, oder unmodifiziert an A vererbt werden:
- Konkretisierung
- Bestandteil, wenn der Eintrag Kontext_abhängig des Bestanteils auf NEIN gesetzt ist
- eines aus Kontext, falls dieser Eintrag nicht auf NEIN gesetzt ist und diese Kanten-Substrukturen zusammen die obligatorische Teilmenge einer Modalitätsmenge bilden

THEN bilde partielle Instanzen ivp(A) von A

Da Strukturrelationen auch Bestandteile zueinander in Beziehung setzen, können diese noch nicht überprüft werden, ebenso die Analysetests. Aus diesem Grund wird hier von einer partiellen Instanz geredet.

IF für ein Konzept A eine partielle Instanz ivp(A) existiert und Instanzen zu allen Bestandteilen existieren, die in der obligatorischen Teilmenge der Modalitätsmenge von A liegen, die ivp(A) zugrunde liegt

THEN bilde Instanzen iv(A) von A

Eine derartige Instanz ist vollständig, kann aber noch um optionale Teile ergänzt werden, gemäß der Regel

IF für ein Konzept A eine Instanz iv(A) existiert und eine Instanz I für eine optionale Kante der Modalitätsmenge von iv(A) existiert

THEN erweitere iv(A) um I zu neuen Instanzen iv(A)

Diese Produktionen sind für jedes Konzept anwendbar, falls gewisse Konsistenzbedingungen, insbesondere die Zyklenfreiheit, für das Netzwerk gelten. Diese Bedingungen sind über die Grade eines Konzepts bzgl. der vier Kantentypen formuliert. Grade und Prioritäten sind rekursiv ausschließlich aufgrund syntaktischer Merkmale der Netzwerksprache definiert /8,11/. Dabei ist die Lage eines Konzepts im Netzwerk bzgl. der Pfade entlang der Kante Konkretisierung von entscheidender Bedeutung.

3. Entwicklungswerkzeuge

Das System ERNEST wurde unter dem Betriebssystem UNIX auf einer VAX in C implementiert. Dabei werden die Netzwerkelemente in C-Strukturen abgelegt. Diese Strukturen können dann binär gespeichert werden. Dadurch wird eine effiziente Ein-/Ausgabe auch bei großen Netzen erreicht. Dies geschieht allerdings um den Preis der flexiblen Netzgröße zur Programmlaufzeit. Die maximale Anzahl der einzelnen Netzwerkkomponenten wird bei der Initialisierung eines Netzes angegeben. Der so erzeugte Speicher-

platz wird dann dynamisch verwaltet. Reicht der Platz nicht mehr aus, so kann das Netz mit einem speziellen Programm erweitert werden. Als Sicherungskopie für die Konzepte dienen Textdateien. Bei Änderung der rechnerinternen Darstellung - die Binärdateien lassen sich dann nicht mehr verwenden - kann das Netz aus den Textdateien wieder erzeugt werden.

Einen Überblick über das gesamte System gibt Bild 2. Die tragende Säule des Netzwerksystems ist die ERNEST-Shell. In sie gelangt der Benutzer durch ein entsprechendes Kommando. Diese spezielle Shell stellt dem Anwender Werkzeuge zum Umgang mit dem semantischen Netz zur Verfügung. Der Name des zu bearbeitenden Netzes wird durch eine globale Variable gesetzt. Auf diese Variable greifen alle Kommandos und Programme zurück.

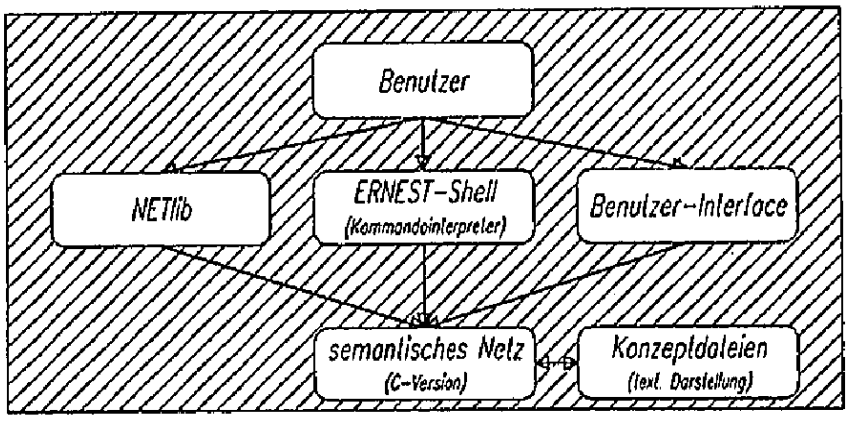

Bild 2: Systemüberblick

```
BEGINNE_KONZEPTDEFINITION( objekt )
   GRADE( 0 0 0 0 )
   PRIORITAET( 3 2 2 0 0 )
   INFORMATION( 3D-Objekt )
   START_KONKRETISIERUNGEN( 2 )
      KONKRETISIERUNG( Oberflaeche )
         DEFINITIONSBEREICH( flaeche )
         AUFSPALTUNG( 1, 10000 )
         AUFSP_BERECHNUNG( flaeche )
         DIMENSION( 1, 1)
      KONKRETISIERUNG( Vertex )
         DEFINITIONSBEREICH( vertex )
         AUFSPALTUNG( 1, 10000 )
         AUFSP_BERECHNUNG( vertex )
         DIMENSION( 1, 1)
   ENDE_KONKRETISIERUNGEN
   START_BEWERTUNG
      NAME( 3d_objekt )
   ENDE_BEWERTUNG
ENDE_KONZEPTDEFINITION( objekt )
```

Bild 3: Konzept 'objekt'

3.1 Benutzer-Interface

Das Benutzer-Interface steht für Benutzer ohne UNIX- und ERNEST-Erfahrung zur Verfügung. Menügesteuert können hierbei die einzelnen Kommandos ausgewählt werden.

3.2 Hauptprogramme und sonstige Kommandos

Für gewisse immer wiederkehrende Aufgaben stehen dem Benutzer C-Hauptprogramme bzw. sonstige Kommandos in Form von Shell-Skripten zur Verfügung. Alle diese Programme und Kommandos arbeiten auf dem semantischen Netz, das vorher mit Hilfe der globalen Variable definiert wurde. Mit einem Kommando kann z.B. der Inhalt des Netzes auf dem Bildschirm ausgegeben werden. Dies ist in drei verschiedenen Arten möglich:

- nur die Konzeptnamen werden dargestellt
- zusätzlich werden die textuelle Information, die Grade und die Prioritäten dargestellt
- die Konzepte werden vollständig ausgegeben

Die Menge der so dargestellten Konzepte läßt sich durch Angabe eines oder mehrerer

regulärer Ausdrücke einschränken. Durch die Aufrufe der entsprechenden Kommandos werden Konsistenztest und Grad- sowie Prioritätsberechnung auf dem semantischen Netz ausgeführt. Der Konsistenztest überprüft dabei die Verzeigerung im semantischen Netz und die Zulässigkeit der Verbindungen entsprechend den in Abschnitt 2.2 erwähnten Einschränkungen.

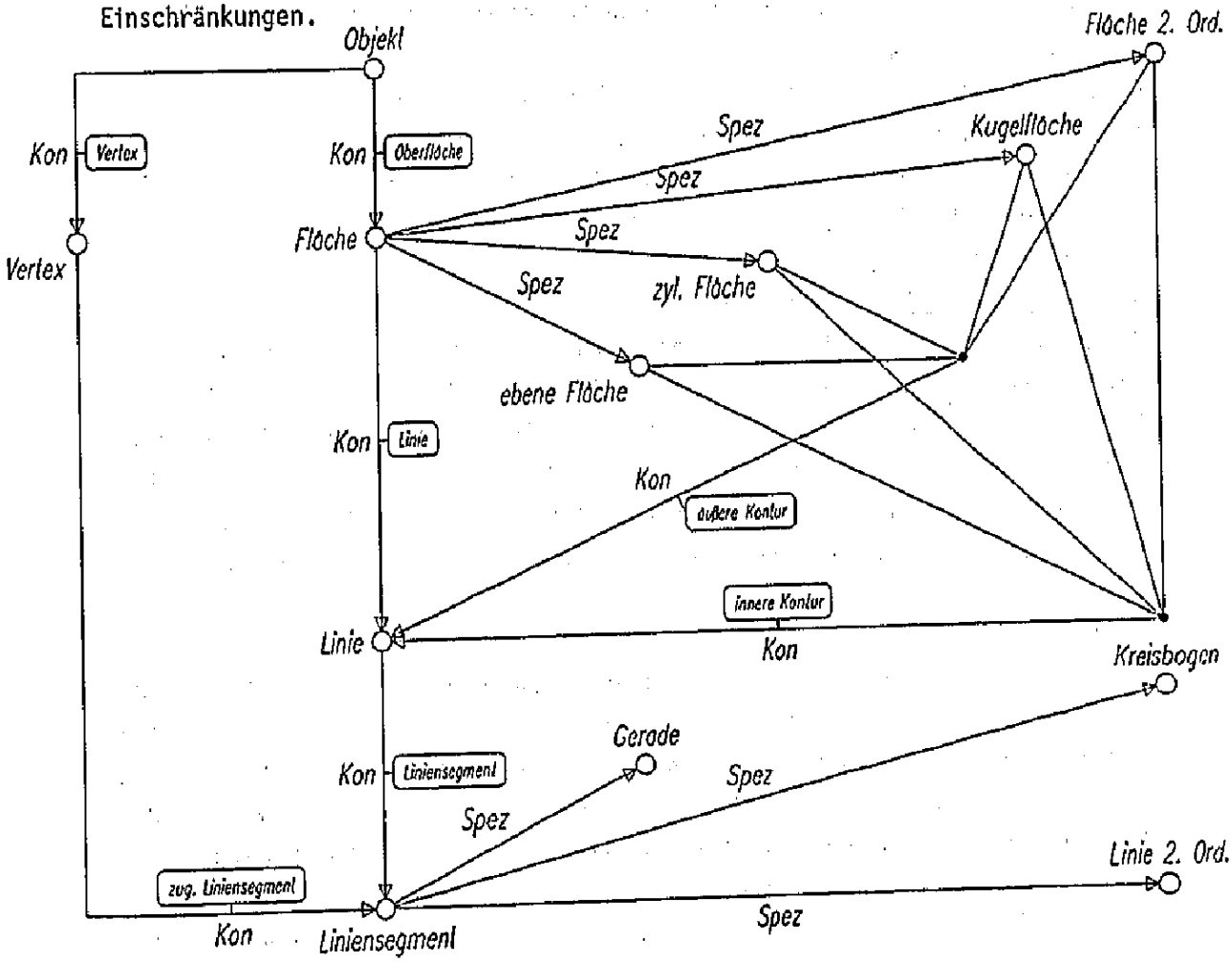

Bild 4: Beispielnetzwerk

Zur weiteren Manipulation der Konzepte stehen dem Benutzer Kommandos zum Kopieren, Löschen und Umbenennen von Konzepten zur Verfügung. Desweiteren können die Konzepte des Netzes durch Aufruf von sextract in Textdateien umgewandelt werden. Bild 3 zeigt den Inhalt der Textdatei für das Konzept 'objekt' aus dem Beispiel von Bild 4.

3.3 Die Netzwerkbibliothek

In der C-Unterprogramm-Bibliothek NETlib sind die Zugriffsroutinen für das semantische Netz abgelegt. Sie gliedern sich in folgende Blöcke:

- Create: Sie fordern Speicherplatz für eine Netzwerkstruktur (Konzept, Teil, Attribut usw.) an.
- Purge: Sie geben den Speicherplatz für eine Netzwerkstruktur wieder frei.
- Insert: Sie fügen in Netzwerkstrukturen Werte bzw. Zeiger ein.

- Delete: Sie löschen in Netzwerkstrukturen Werte bzw. Zeiger.
- Replace: Sie ersetzen in Netzwerkstrukturen Werte bzw. Zeiger.
- Get: Sie lesen aus Netzwerkstrukturen Werte bzw. Zeiger.
- sonstige: Dieser Block enthält die Unterprogramme zum Berechnen der Grade und der Prioritäten, zum Kopieren und Darstellen von Konzepten und zur Ein-/Ausgabe des gesamten Netzes.

Durch diese C-Unterprogramme sind die eigentlichen C-Strukturen, in denen die einzelnen Netzwerkelemente abgelegt sind, für den Anwendungsprogrammierer verdeckt. Bei Änderung dieser Strukturen müssen so nur die Zugriffsroutinen ebenfalls geändert werden, nicht aber die Programme der einzelnen Anwender.

3.4 Der wissensbasierte Netzwerkeditor

Zur interaktiven Manipulation des semantischen Netzes existiert ein textorientierter Editor. Unter Verwendung von Fenstertechnik werden dabei die im Netz vorhandenen Einträge dargestellt. Diese können dann je nach Eintrag gelöscht, modifiziert und erweitert werden. Der Editor sorgt automatisch dafür, daß für neu eingetragene bzw. gelöschte Kanten die Rückverzeigerung ebenfalls entsprechend modifiziert wird. ERNEST- und UNIX-Kommandos können aus dem Editor heraus abgesetzt werden.

3.5 Graphik

Der oben beschriebene textorientierte Editor kann nur die einzelnen Einträge des semantischen Netzes aufzeigen. Es ist aber vorteilhaft, auch die Netzwerkstruktur darzustellen. Deshalb soll eine Graphikerweiterung des Editors entwickelt werden. Das Erzeugen einer Graphik ist aber ein Problem, da ein Netz mit einigen hundert Konzepten und mehreren hundert Kanten nicht ohne weiteres in übersichtlicher Form dargestellt werden kann. Gedacht ist daher an Teilansichten, die immer nur zwei Kantentypen enthalten. Da jedes Konzept in seinem Grad Information über die Lage innerhalb des Netzwerks bezüglich der vier Kantentypen besitzt, lassen sich so Konzepte eindeutig in einem zweidimensionalen Koordinatensystem (die Koordinaten stellen die zwei zu betrachtenden Kantentypen dar) ablegen. So kann dann unter Angabe eines 'Startkonzepts' und einer maximalen Pfadlänge automatisch eine Teilansicht generiert werden. Eine ebenfalls wünschenswerte Komplettansicht kann nur interaktiv erzeugt werden. Dabei erhält jedes Konzept und jede Kante graphische Parameter, die eine graphische Darstellung erlauben. Bild 4 zeigt eine solche Komplettansicht für ein kleines Netz zur Beschreibung von allgemeinen 3D-Objekten anhand von Oberflächen und Vertices.

4. Analysewerkzeuge

Die leichte Handhabung von kodiertem "Wissen" für Analyseprozesse war ein grundlegender Gedanke bei der Entwicklung des Systems ERNEST. Die im folgenden vorgestellten Werkzeuge sind dann anwendbar falls ein Netzwerk für eine spezielle Anwendung

erfolgreich auf seine Konsistenz getestet wurde. Sollen diese Werkzeuge keine Verwendung finden, so kann eine Konzeptdefinition auch ohne die in Bild 1 mit * gekennzeichneten Einträge erfolgen. Eine Konsistenzprüfung erübrigt sich in diesem Fall.

4.1 Analysevorbereitung

Für die einfache Erstellung und Modifikation und für die Übersichtlichkeit eines semantischen Netzes sollte die Wissensrepräsentation möglichst redundanzfrei sein. Die implizite Vererbung von Bestandteilen, Konkretisierungen, Attributen, Analyseparametern, Strukturrelationen und Analysetests bietet deshalb eine Möglichkeit, Begriffe eines Problemkreises komprimiert zu repräsentieren. Bei der Analyse hingegen steht eine möglichst effiziente Nutzung des Wissens im Vordergrund, die zur Redundanzfreiheit im Widerspruch steht. Deshalb wird in einem Analysevorbereitungslauf das Netz so aufbereitet und ergänzt, daß der Kontrolle alle Information, die sie benötigt und die vorab berechnet werden kann, zur Verfügung steht.
Mit Hilfe der Prozedur inhereted_slots werden für alle Konzepte die Substrukturen berechnet, die aus Generalisierungen geerbt werden. Es werden Zeiger auf diese im jeweiligen Konzept eingetragen, so daß die Substrukturen nicht dupliziert werden müssen und der zusätzliche Speicheraufwand sich in Grenzen hält. Der Effizienzgewinn bei der Analyse ist offensichtlich, da für Zugriffe auf eine Substruktur, nur noch das betrachtete Konzept in Frage kommt. Die höchst aufwendige Untersuchung, woher eine Substruktur geerbt wurde, entfällt. Darüber hinaus müssen bei der direkten Instantiierung eines Konzepts (siehe /8/) keine partiellen Instanzen der Generalisierungen angelegt werden, die Substrukturen in dieses Konzept vererben, da diese nun auch im betrachteten Konzept referiert sind. Dieses bedeutet eine erhebliche Rechenzeit- und Speicherplatzersparnis.
Für die Werteberechnungsfunktion der Attribute und Analyseparameter sind auch selbstreferenzierende Argumente zugelassen. Das heißt für die Werteberechnung können neben berechneten Werten aus Attributen und Analyseparametern von Konkretisierungen, Bestandteilen und von Konzepten, die als Kontext des betrachteten Konzepts genannt sind, auch bereits berechnete Werte aus Attributen und Analyseparametern des aktuellen Konzepts verwendet werden. Dazu ist aber erforderlich, daß zum einen die Selbstreferenzen widerspruchsfrei sind und zum anderen eine gewisse Reihenfolge bei der Berechnung dieser Substrukturen eingehalten wird. Mit test_arguments werden Argumente von Attributen und Analyseparametern auf unauflösliche Referenzen hin untersucht und evtl. Fehler ausgegeben. Treten keine Fehler auf, so kann mit create_float_chart der sog. Konzeptflußgraph berechnet werden. Dieser gibt zum einen die Reihenfolge für die Werteberechnung an und zum anderen stellt er die Selbstreferenzen dar. Dazu wird für jedes Attribut und für jeden Analyseparameter ein Knoten erstellt und gemäß der Selbstreferenzen doppelt verzeigert ("referiert" und "referiert-von"). Zusätzlich wird jeder Knoten einer Ebene zugeordnet, wobei diese die Berechnungshierarchie widerspiegeln. Knoten der Ebene 0 sind Attribute und Analyseparameter

ohne Selbstreferenzen, während Knoten der Ebene n mindestens eine Referenz zu einem Knoten der Ebene n-1 und keine zu einem Knoten der Ebenen >=n besitzen. Für die Berechnung geht man die Ebenen von unten nach oben sukzessive durch und kann somit die Werteberechnung ohne großen Verwaltungsaufwand während der Analyse durchführen. Mit create_initial_search_graph wird für jedes Konzept der sog. Netzflußgraph berechnet werden. Dieser gibt alle möglichen Pfade für die Instanziierung eines Konzepts an, die sich aus der Produktion zur Instanziierung ergeben. Diese ebenfalls im voraus zu berechnende Information trägt zur Effizienz der Analyse bei, da der Suchgraph für einen Analyseprozeß eng an den Netzflußgraphen gebunden ist. Er unterscheidet sich in seiner Struktur nur durch die Anzahl der von einem Knoten ausgehenden Nachfolger und durch zusätzliche Knoteninformationen.

4.2 Kontrollalgorithmen

Die in Abschnitt 2.2 beschriebenen Produktionen zur Instantiierung von Konzepten legen den Kern einer jeden Verarbeitungsstrategie fest. Ist ein Analyseziel (i.a. eines oder mehrere Konzepte) bekannt, so kann zusätzlich der Netzflußgraph dieser Konzepte betrachtet werden. Die einfachste Strategie geht von diesen Zielkonzepten aus. Mit Hilfe der Netzflußgraphen wird eine top-down Suche durchgeführt und anschließend werden die Konzepte des Netzflußgraphen bottom-up instantiiert. Jeder Knoten des so entstehenden Suchbaumes enthält ein expandiertes Modell eines Zielkonzepts, welches den augenblicklichen Stand der top-down Suche und bottom-up Instantiierung von der Wurzel des Suchbaumes bis zu diesem Knoten widerspiegelt. Gemäß den Bewertungen von einzelnen Instanzen und einer Schätzung für die Bewertung des Suchbaumknotens anhand eines Zielkonzepts kann ein derartiger Analyseprozeß sehr effizient mit dem A*-Algorithmus gesteuert werden /8/. Neben diesem Kontrollalgorithmus steht ein weiterer zur Verfügung. Dieser arbeitet flexibel bottom-up und top-down, wobei eine gegebene Wissensbasis sukzessive an bereits detektierte Fakten angepaßt wird. Dies geschieht während des Analyseprozesses durch den Aufbau von modifizierten Konzepten. Dieser Algorithmus wird im folgenden zusammen mit den Funktionen create_instance und create_td_modconcept vorgestellt. Die Stratgie beruht auf einer Modifikation des A*-Algorithmus, wobei mehrstellige Bewertungsfunktionen zugelassen sind /11/.

Bei der Expansion eines Konzepts im aktuellen Knoten werden gemäß der Produktion zur Instanziierung die Konkretisierungen und die Bestandteile in die strukturelle Analyse miteinbezogen. Dabei werden diese an den aktuellen Stand der Analyse mittels einer invertierten Berechnung von Attributen, Analyseparametern, Strukturrelationen und Analysetests angepaßt, indem mit Hilfe der Funktion create_td_modkonzept (Erzeuge top-down modifizierte(s) Konzept(e)) Konzepte modifiziert werden. Dadurch wird die Wissensbasis lokal für den aktuellen Knoten gemäß der top-down Erwartungen aus der strukturellen Analyse und gemäß bottom-up Hypothesen aus der Segmentierung des Bild/- oder Sprachsignals eingeschränkt. Diese sehr komplexe Funktion benötigt als

Parameter den aktuellen Knoten und das zu modifizierende Konzept oder ein bereits modifiziertes Konzept, das weiter eingeschränkt werden soll. Sie gibt das (die) erzeugte(n) modifizierte(n) Konzept(e) zurück, wobei konkurrierende modifizierte Konzepte sich aus einer nicht unbedingt eindeutigen invertierten Berechnung von Attributen und Analyseparametern ergeben. Zuerst wird die invertierte Berechnung von Strukturrelationen und Analysetests durchgeführt. Diese dient zur Einschränkung des Wertebereiches von Attributen und Analyseparametern. So kann eine Strukturrelation, die überprüft, ob der Wert eines Attributs größer als der Wert eines anderen ist, invertiert werden, indem der Wertebereich für ein Attribut eingeschränkt wird, falls der Wert des anderen vorliegt. Dadurch kann die Zahl möglicher Ausprägungen für das modifizierte Konzept verringert werden. Daraufhin wird die invertierte Berechnung der Attribute und der Analyseparameter angestoßen. Hierbei werden Vorerwartungen und konkrete Daten ausgenuzt, um einen möglichst guten Vorhersagewert für diese Substrukturen zu erhalten. Werden mehrere konkurrierende Werte berechnet, so wird das entsprechende modifizierte Konzept vervielfältigt und die weitere Verarbeitung wird für alle analog durchgeführt. Diese besteht dann aus der Überprüfung der Restriktionen für Attribute und Analyseparameter. Dabei wird der Vorhersagewert auf seine Zulässigkeit bezüglich des vorgegeben bzw. eingeschränkten Wertebereichs hin untersucht und bewertet. Abschließend wird für das modifizierte Konzept die Bewertungsfunktion angestoßen.

Für die Instantiierung eines (evtl. modifizierten) Konzepts steht dem Benutzer ebenfalls eine kompakte Funktion (create_instance) zur Verfügung, die die Realisierung eines Kontrollalgorithmus wesentlich erleichtert. Übergabeparameter sind der aktuelle Knoten und das zu instanziiernde (evtl. modifizierte) Konzept. Analog zur Modifikation von Konzepten können aufgrund einer nicht unbedingt eindeutigen Berechnung von Attributen und Analyseparametern mehrere konkurrierende Instanzen erzeugt werden. Folgende Schritte sind dazu nötig:

- Erzeugung einer Rohinstanz, in der nur die Konkretisierungen und die Bestandteile gemäß dem aktuellen Knoten verzeigert sind.
- Überprüfung und Bewertung der Restriktionen für Konkretisierungen und Bestandteile.
- Berechnung der Attribute und der Analyseparameter mit Hilfe des Konzeptflußgraphen. Generiere bei konkurrierenden Werten konkurrierende Instanzen.
- Überprüfung und Bewertung der Restriktionen für Attribute und Analyseparameter.
- Berechnung der Struktur- und der Analyserelationen.
- Berechnung der Instanzbewertung.

Wie man sieht stellen die beiden oben beschriebenen Funktionen mächtige Werkzeuge für die Analyse dar, da sie unabhängig vom konkreten Kontrollalgorithmus die wesentlichen Schritte der Analyse durchführen, nämlich lokale Anpassung der Wissensbasis und Instanziierung von Konzepten.

5. Akquisitionswerkzeuge

Die Fähigkeit zur Wissensakquisition ist für moderne Bildanalysesysteme von großer Bedeutung. Obwohl diese Systeme selbst modular konzipiert sind, ist eine Anpassung an ein neues Aufgabengebiet ein zeitaufwendiger Vorgang. Der Flaschenhals ist hierbei die Anpassung der Wissensbasis. Grundsätzlich kann man zwei Methoden der Wissensakquisition unterscheiden: den manuellen Ansatz (die Arbeit liegt beim Entwickler) und den automatischen Ansatz (die Arbeit liegt bei der Maschine) Außerdem existieren eine Reihe von Zwischenlösungen /1/.
Bis jetzt haben wir den manuellen Ansatz verfolgt. Der Systementwickler konnte mit dem in Abschnitt 3 vorgestellten Netzwerkeditor den deklarativen Teil der Wissensbasis erzeugen. Unser Ziel ist es jetzt, einen zum größtenteil automatischen Wissensakquisitionsalgorithmus zu entwickeln. Dadurch sind dann auch Personen, die keine oder nur wenig Erfahrung im Umgang mit semantischen Netzen haben, in der Lage, das Bildanalysesystem an eine neue Problemstellung anzupassen.

5.1 Netzwerkelemente für die Akquisition

Die schraffierten Blöcke in Bild 1 zeigen die Netzwerkelemente, die von der Akquisition verwendet werden. Wenn ein neues Konzept durch einen Akquisitionslauf aus einem sogenannten Metakonzept erzeugt wird, so werden diese beiden Konzepte durch eine Kante mit dem Namen **Modell** verbunden. Die inverse Kante heißt **Modell_von**. Ein Konzept der Wissensbasis kann natürlich nur aus einem Metakonzept modelliert worden sein, während ein Metakonzept mehrere modellierte Modellkonzepte haben kann. Entlang der Kante Modell existiert keine Vererbung. Statt dessen enthält ein Metakonzept quasi eine Konstruktionsbeschreibung für Modellkonzepte.
Während der Akquisition kann es notwendig sein, Strukturen des Metakonzepts aufzuspalten. So besitzt das Metakonzept OBJEKT aus Bild 4 die Konkretisierung FLÄCHE. Ein konkretes Modellkonzept z.B. eines Würfels hat aber mehrere Flächen. Deshalb wird die in diesem Fall notwendige Aufspaltung durch ein vorgegebenes Intervall beschränkt. Der Eintrag **Aufspaltung** enthält die Intervallgrenzen und der Eintrag **Aufsp.-Berechnung** verzeigert zu einer Prozedur, die die notwendige Anzahl an Aufspaltungen berechnet. Das gleiche Problem kann bei der Dimension auftreten. Verschiedene Modellkonzepte eines Metakonzepts können unterschiedliche Dimensionen in ihren Strukturen besitzen. Der Eintrag **Dimension** enthält wieder die Intervallgrenzen und der Eintrag **Dim._Berechnung** referiert die Berechnungsprozedur. Der Eintrag **MModifiziert** wird verwendet, wenn eine Aufspaltung stattgefunden hat. In diesem Fall wird dort die Rolle der jeweiligen Netzwerkstruktur im Metakonzept eingetragen. Wenn keine Aufspaltung nötig war, wird MModifiziert auf NEIN gesetzt. Der Eintrag **Argumenttest** referiert grundsätzlich auf eine Prozedur, die die Existenz von Argumenten für Analyseprozeduren prüfen. Der Eintrag **Defaultberechnung** verzeigert auf eine Prozedur, die Defaultwerte während der Akquisition berechnet.

5.2 Metawissen

Die Wissensbasis des Akquisitionsprozesses enthält das Metawissen. Es teilt sich in deklaratives und prozedurales Metawissen. Der deklarative Teil ist das Metamodell; ein semantisches Netz, das aus den Metakonzepten gebildet wird. Dieses Metamodell stellt eine allgemeine Beschreibung für ein bestimmtes Aufgabengebiet dar. Bild 4 zeigt ein solches Metamodell für die Akquisition von 3D-Objekten anhand von Oberflächenbeschreibungen.

Der prozedurale Teil besteht aus den Prozeduren, die im Metamodell über die Einträge Argumenttest, Diff.-Berechnung, Dim.-Berechnung und Defaultberechung referiert werden. Außerdem gehören noch die zum Vergleich von Zwischenergebnissen benötigten Regeln zum prozeduralen Metawissen. Der Akquisitionsalgorithmus arbeitet mit beiden Teilen des Metawissens. Algorithmus und Metawissen sind aber insoweit unabhängig, als daß der Algorithmus auch mit einem anderen Metawissen arbeiten kann.

5.3 Der Akquisitionsalgorithmus

Der Akquisitionsalgorithmus unterteilt sich in zwei Schritte. Im ersten Schritt muß eine Beobachtung in ein semantischen Netz transferiert werden. Diese Beobachtung kann z.B. eine Aufnahme des zu lernenden Objekts sein oder aber seine Beschreibung in Form von CAD-Daten. Der Algorithmus benötigt zwei Parameter. Der erste ist der Name des Metakonzepts, das den Ablauf steuert. Der zweite ist der Name des neu zu erzeugenden Modellkonzepts.

In der ersten Phase von Schritt 1 wird ein Hüllkonzept angelegt. Hüllkonzept bedeutet, daß zwar die Konzepte, die Substrukturen und die Kanten erzeugt werden, daß aber die Einträge innerhalb der Konzepte noch nicht gefüllt sind. In diese Phase fallen also auch die Aufspaltung und die Berechnung der Dimensionen.

Die in der zweiten Phase zu erfüllende Aufgabe ist es, die Existenz der Argumente der Analyseprozeduren zu testen. Dies geschieht, indem die durch die Einträge mit dem Namen Argumenttest referierten Prozeduren aufgerufen werden. Ein Argument besteht aus maximal zwei funktionalen Rollen. Solche aus dem Metakonzept übernommenen Rollen können aufgrund von Aufspaltungen nicht mehr vorhanden sein. Wenn nämlich eine Netzwerkstruktur des Metakonzepts aufgespalten wird, bekommen die neuen Strukturen unterschiedliche Rollen. Dabei kann die Rolle aus dem Metakonzept nicht übernommen werden. In diesem Fall müssen die neuen Rollennamen für die jeweiligen Argumente ermittelt werden.

In der dritten Phase von Schritt 1 werden dann noch die Defaultwerte berechnet. Für den späteren Analyselauf kann es sinnvoll sein, wenn auch der Definitionsbereich von Attributen und Analyseparametern eingeschränkt wird. Auch dieses wird für die jeweiligen Netzwerkstrukturen in der dritten Phase durchgeführt.

Die in Schritt 1 erzeugten Konzepte stellen die Zwischenergebnisse des Akquisitionsprozesses dar. Im zweiten Schritt werden nun diese Konzepte regelbasiert verglichen. Dazu werden zunächst alle vorliegenden Ergebnisse von Schritt 1 vereinigt. So ge-

langt man zu einer vollständigen Beschreibung aller verwendeten Beobachtungen. Abhängig von der Intention - Korrektur von Segmentationsfehlern, Lernen von Klassenbeschreibungen oder Lernen von Beschreibungen spezieller Objekte - werden nun unterschiedliche Regeln auf diese Beschreibung angewendet. Beispiele für diese Regeln finden sich in /13/. Falls Negativbeispiele existieren, können diese zur anschließenden Verifikation der Ergebnisse aus Schritt 2 verwendet werden. Die so gewonnenen Beschreibungen in Form von semantischen Netzen bilden dann den deklarativen Teil der Wissenbasis für den Analyselauf.

6. Zusammenfassung

Das System ERNEST stellt einen Rahmen für unterschiedliche Anwendungen im Bereich der Bildanalyse dar. Den Kern bildet ein semantisches Netzwerk, das deklarative und prozedurale sowie symbolische und quantitative Möglichkeiten umfaßt. Es bietet die Möglichkeit "Wissen" von Bildmatrizen ausgehend bis zu einer speziellen anwendungsorientierten Zielebene in einer homogenen Wissensbasis zu integrieren. ERNEST stellt auch Kontrollalgorithmen und Funktionen zum Aufbau solcher Algorithmen zur Verfügung. ERNEST ist in C unter UNIX realisiert und somit leicht portierbar.

Literatur:

/1/ Niemann, H.: Pattern Analysis. Springer-Verlag Berlin Heidelberg New York, 1981

/2/ Buchanan,E.H., Shortliffe,E.H.: Rule-based expert systems. Addison Wesley, 1984

/3/ Hanson,A.R., Riseman,E.M. (eds.): Computer Vision Systems. Academic Press, 1978

/4/ Ballard,D.H., Brown,C.M.: Computer Vision. Prentice Hall, Englewood Cliffs, New Jersey 1982

/5/ Perkins, W.A.: INSPECTOR: A Computer Vision System that Learns to Inspect Parts. IEEE PAMI, Vol.5, No.6, 1986, 584-592

/6/ Brachman,R.J.: On the Epistemological Status of Semantic Networks. in /7/, 3-50

/7/ Findler, N.V. (ed.): Associative Networks. Academic Press, New York 1979

/8/ Sagerer,G.: Darstellung und Nutzung von Expertenwissen für ein Bildanalysesystem. Springer-Verlag Berlin Heidelberg New York Tokyo, IFB 104, 1985

/9/ Brachman,R.J., Schmolze,J.G.: An Overview of the KL-ONE Knowledge Representation System. in Cognitive Science 9, 171-216, 1985

/10/ Levesque,H., Mylopoulos,J.: A Procedural Semantic for Semantic Networks. in /7/, 93-121

/11/ Sagerer,G., Kummert,F., Schukat-Talamazzini,E.G.: Ein flexibler Kontrollalgorithmus für ein sprachverstehendes System. DFG-Kolloquium Modelle und Strukturen, Hamburg April 1987

/12/ Schröder,S., Sagerer,G.: An Associative Network as System Shell for Knowledge Based Image Understanding, Proc. CAIP 87, Wismar 1987

/13/ Dietterich, Michalsky,: Inductive Learning of Structural Descriptions AI, vol.16 1981, 257-294