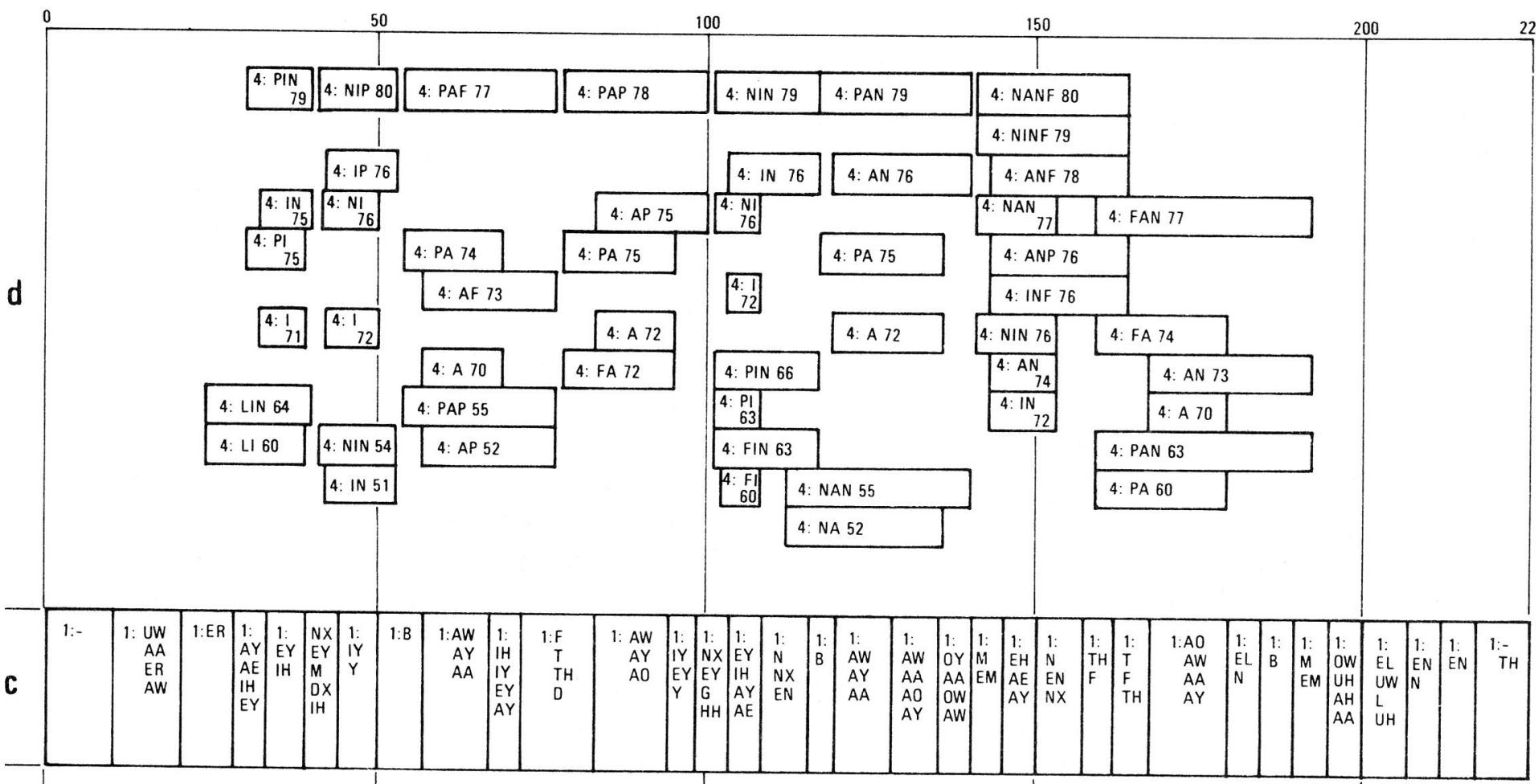
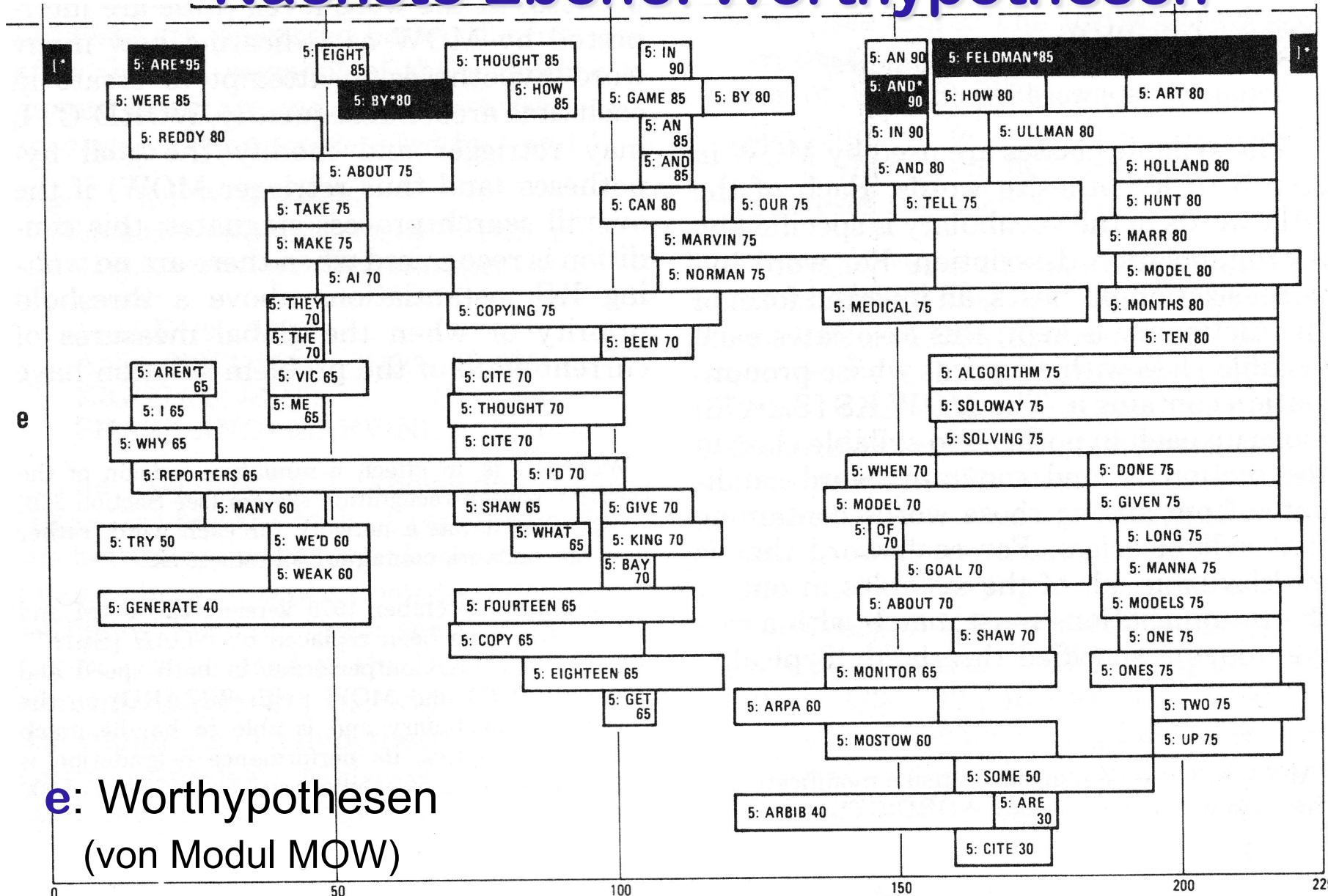


Nächste Ebene: Silbenhypothesen



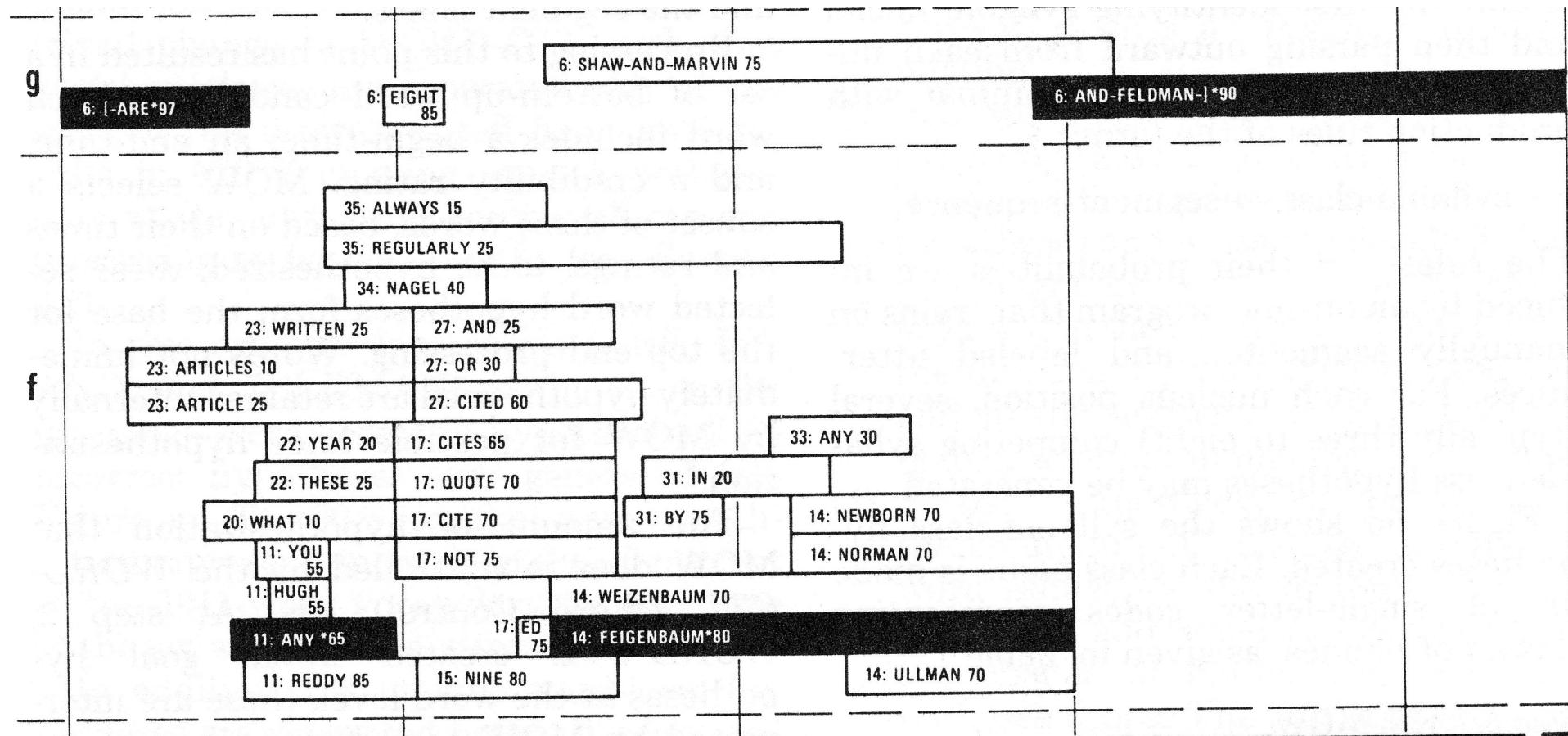
d: Silbenhypothesen

Nächste Ebene: Worthypothesen

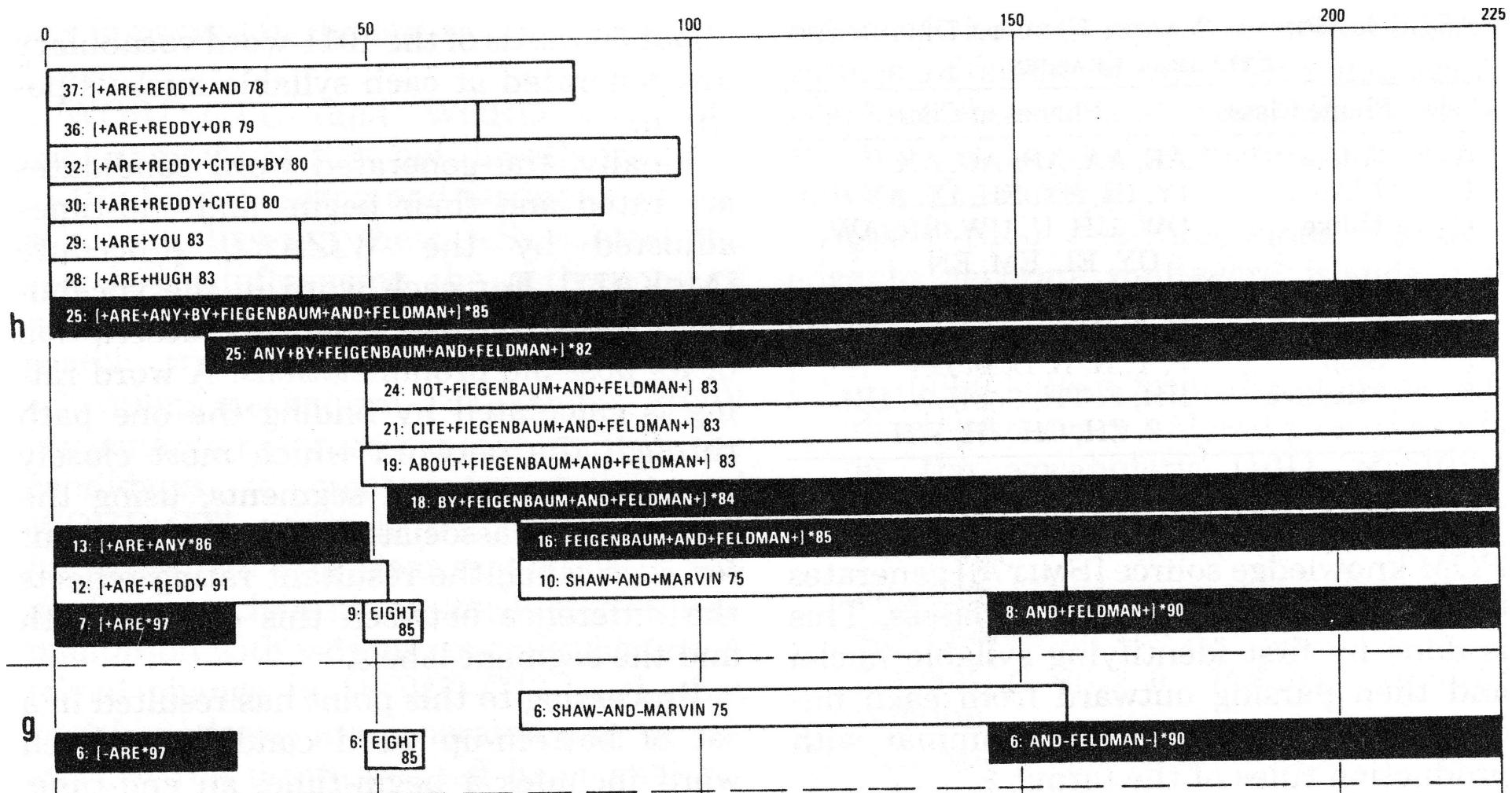


8. Umgebungsdateninterpretation 8.3 Semantische Kartierung

Nächste Ebene: Worthypothesen & -folgen



Oberste Ebene: Phrasenhypothesen



h: Phrasenhypothesen

Die Hearsay-Lektion

- Blackboard-Architektur erlaubt flexible Sprachsignalverarbeitung (=Sensordatenverarbeitung), bei der Information aus ganz unterschiedlichen Wissensebenen zusammenkommt, ohne a-priori-Verarbeitungsreihenfolge!

Aber:

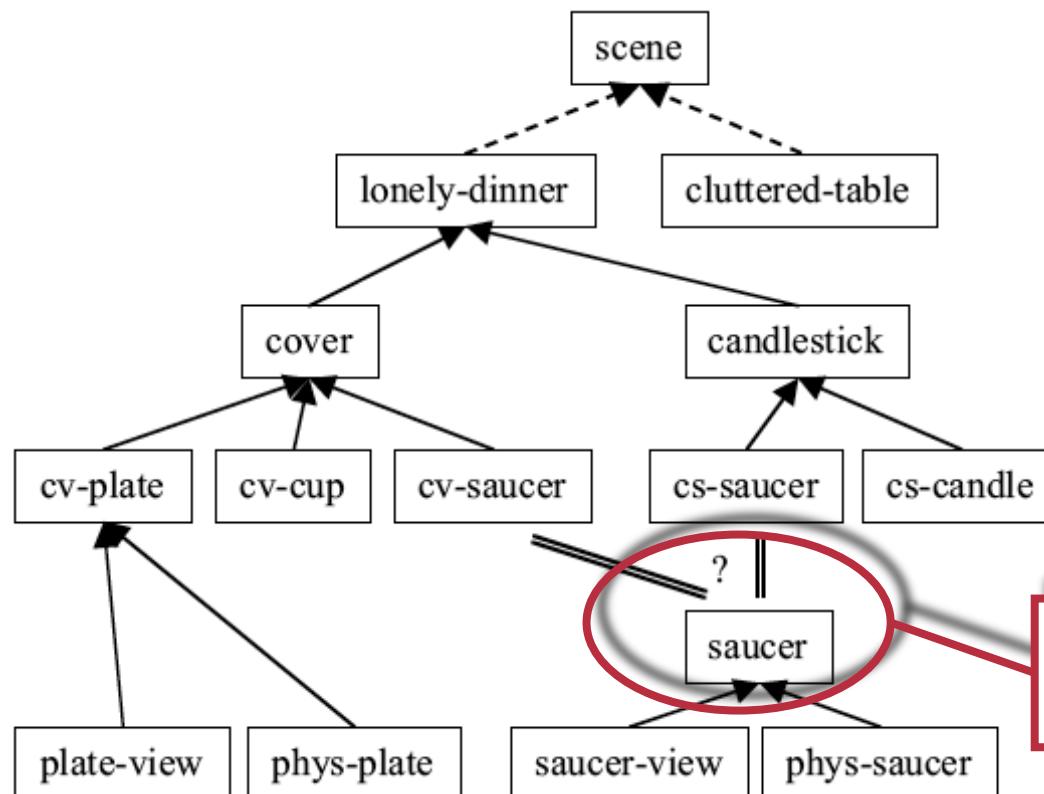
- Die Suchräume der einzelnen Module sind groß; der des Gesamtsystems ist riesig! Skalierung auf höhere Systemleistung geht nur bei effizienter Suchraumkontrolle
- Erweiterung auf größeren Funktionsumfang (Lexikon, Grammatik) ist ebenfalls nicht ohne Weiteres möglich

Systemleistung Hearsay II zum Abschluss (1976/77):

1 Sprecher, 1011 Wörter, 9% *sentence semantic error* (falscher Satzsinn),
19% *sentence err.* (Fehler in einzelnen Wörtern, aber ggf. korrekter Satzsinn)

Zurück zu semantischen Karten

Blackboard-Architektur bietet einen Lösungsansatz zur Strukturierung der Kontrolle bei Szeneninterpretation (solange niemand eine bessere Lösung hat)!



Beispiel (Neumann/Möller):
Auswahl unter möglichen
Aggregat-Instanzierungen
entspricht etwa der Auswahl
möglicher Wortfolgen aus
Silbenfolgen

Baustellen bei Neumann/Möller

- Es gibt keine **temporären individuellen Objekte und Aggregate** („meine (aktuelle) Tasse“, „mein (aktuelles) Gedeck“)
- Die **Wahl möglicher Schritte der Szeneninterpretation** (Aggregat-Instanzierung, Instanz-Verfeinerung, Instanz-Mischung) und ihrer jeweils aktuellen Anwendung ist weitgehend offen (vorgeschlagen: probabilistischer Ansatz; Blackboard-Implementierung liegt nahe)
- Es wird nichts zum **Tracking/Objektverankerung** gesagt: Wie sorgt man für die Kontinuität der Identität von Objekten über die Zeit, auch wenn sie zwischenzeitlich aus der Sensorik verschwinden? (s. Folie 401)

→ BA-Arbeit Claudia Schulz, 2011

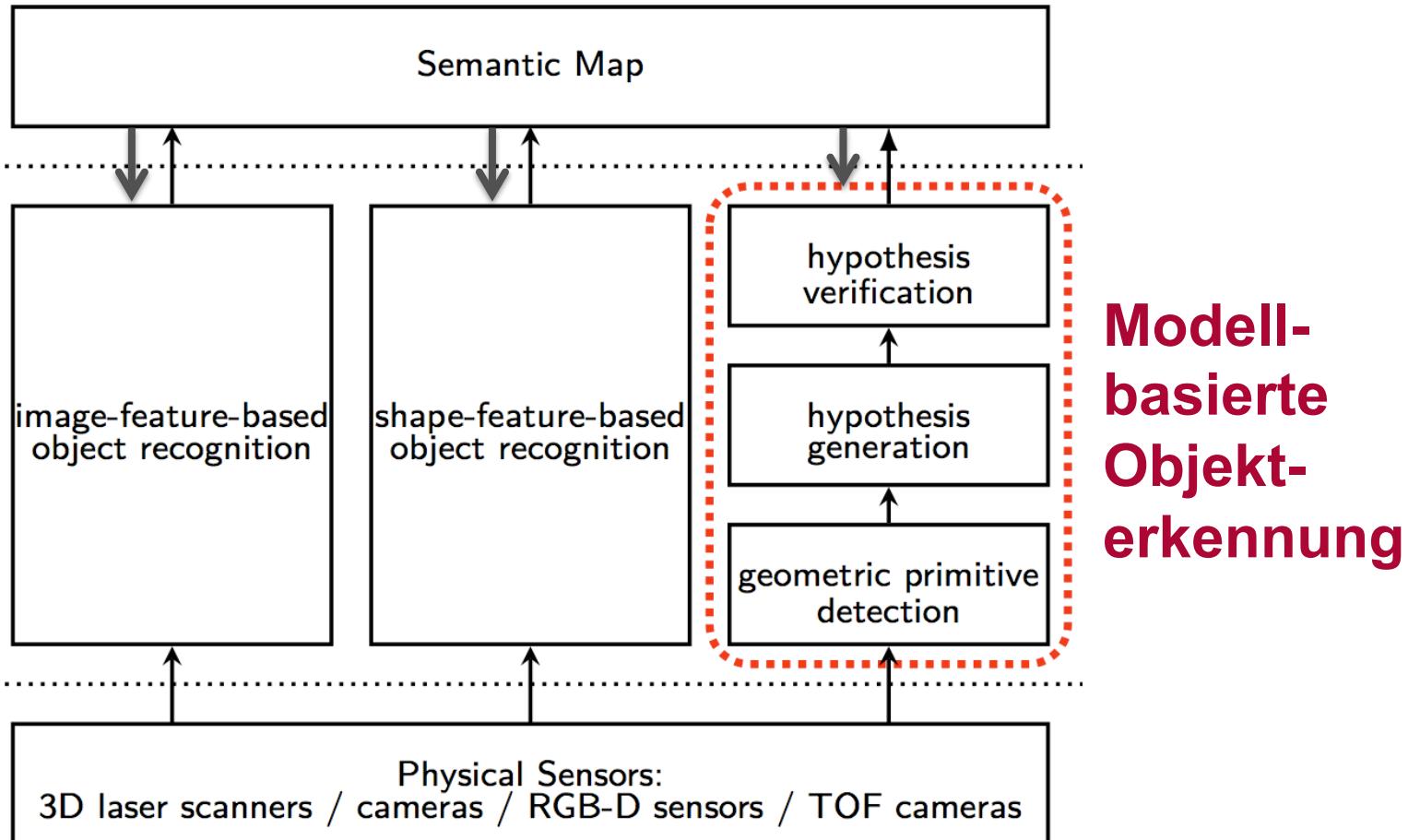
Semantische Kartierung auf 3D-Punktwolken

- **Gegeben:** 3D Punktwolke
 - ... wie von Laserscanner,
ToF-Kamera, Kinect, ...
 - ... normalerweise aus vielen
Scans registriert (autom.)
- **Gegeben:** CAD-Modell
einiger Objekte
 - ... vom Hersteller,
Google 3D Warehouse, ...
- **Gegeben:** Modell geometrischer
Objektkonstituenten
 - ... handgemacht (derzeit) oder aus CAD-Modell (in Zukunft)
- **Finde:** Vorkommen der Objekte in den Daten



Architekturannahme

Andere Objekterkennungs-
methoden dürfen/sollten
zusätzlich existieren!



Schritt I: Erkenne Geometrische Primitive

- Erkenne Primitive (Ebene, Zylinder, Kugel) aus Punktnormalen in dichten Teil-Wolken
- Möbel: Beschränkung auf planare Anteile (unabhängig von Orientierung!)

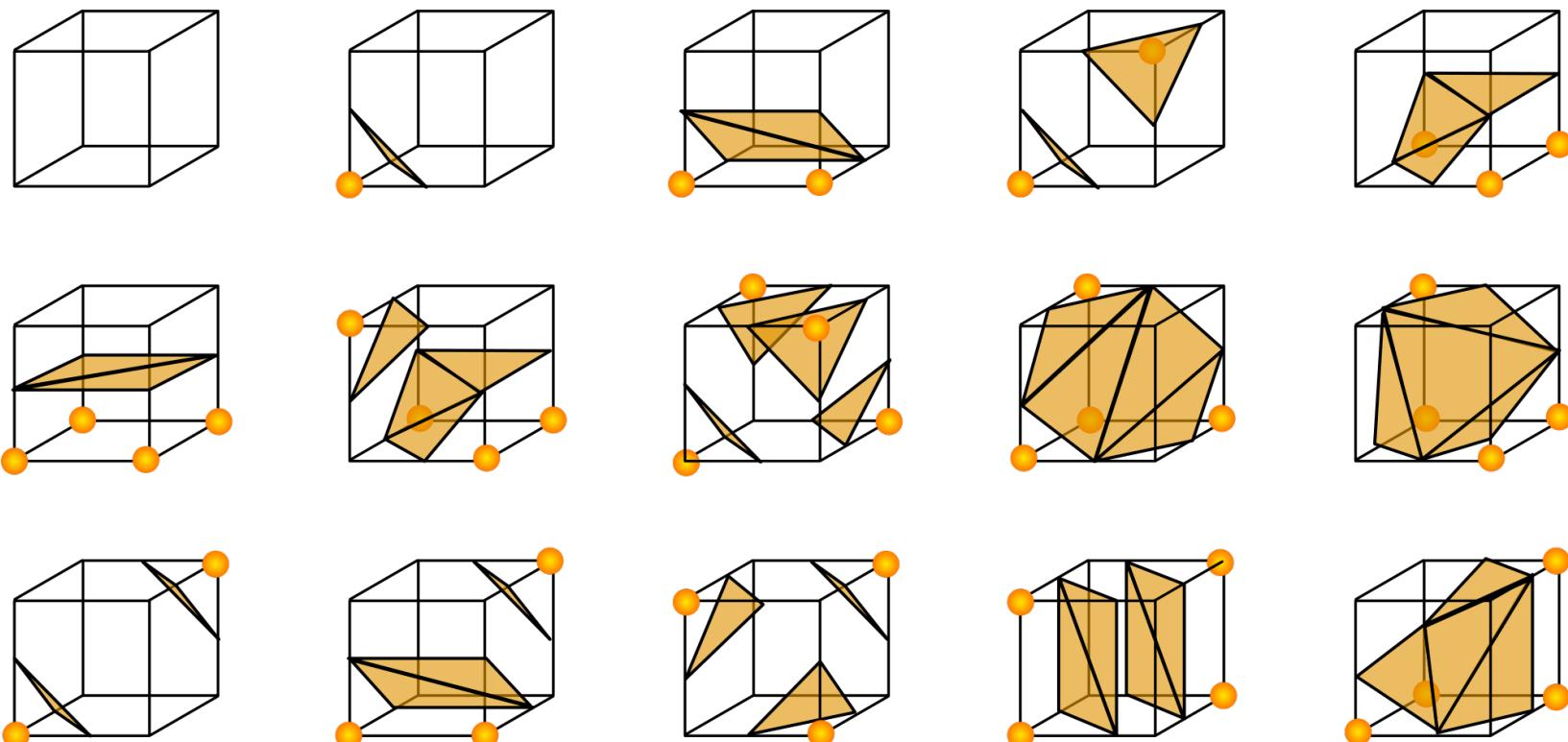


- Erzeuge Dreiecks-Mesh durch optimierte Marching-Cubes-Implementierung
- Region-growing entlang homogener Dreiecks-Normalen



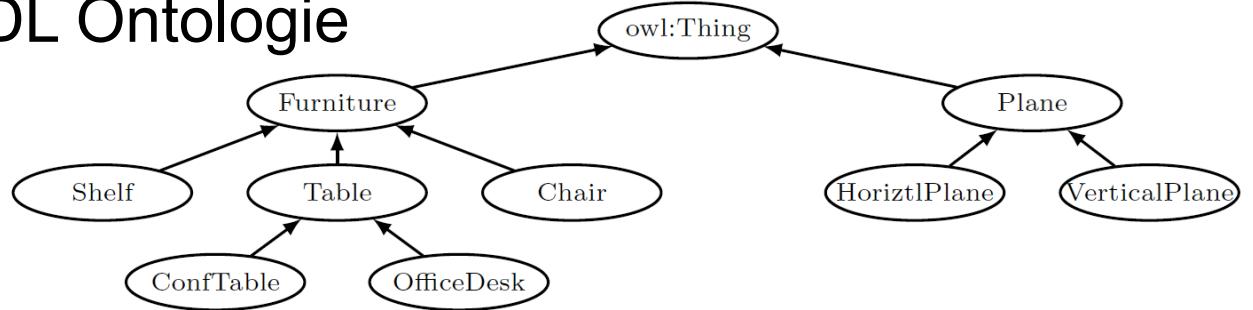
Marching Cubes ...

... ist ein Standard-Algorithmus aus der CG zur Erzeugung polygonaler Darstellungen aus Voxeln mit Punkten



Schritt II: Erzeuge Objekthypothesen

- ... aufgrund OWL-DL Ontologie



- Prüfe Relationen aufgrund Sensordaten (Größe, Lage); kombiniere SWRL Regeln mit Ontologie

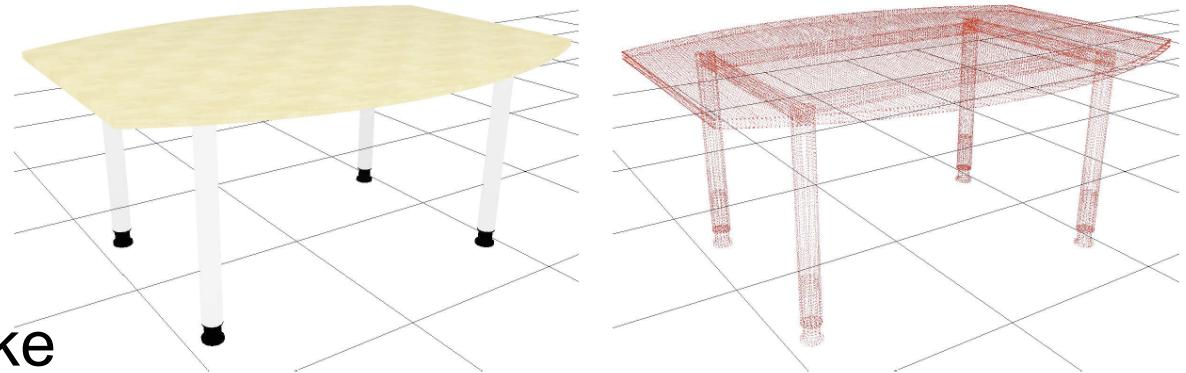
*Table(?p) ← HorizontalPlane(?p) ∧ hasSize(?p,?s) ∧
swrlb : greaterThan(?s,1.0) ∧ hasPosY(?p,?h) ∧
swrlb : greaterThan(?h,0.65) ∧ swrlb : lessThan(?h,0.85)*

- Berechne Objektpose-Hypothese (Flächennormalen, PCA, ...)

Schritt III: Verifizierte Objekthypothesen

Grundidee

- Sample CAD-Modell in 3D Punktfolge
- Registriere Punktfolge mit 3D Messpunktfolge an hypothetischer Pose (mit ICP)
- Akzeptiere Objekthypothese falls Registrierungsfehler unter Schwellwert



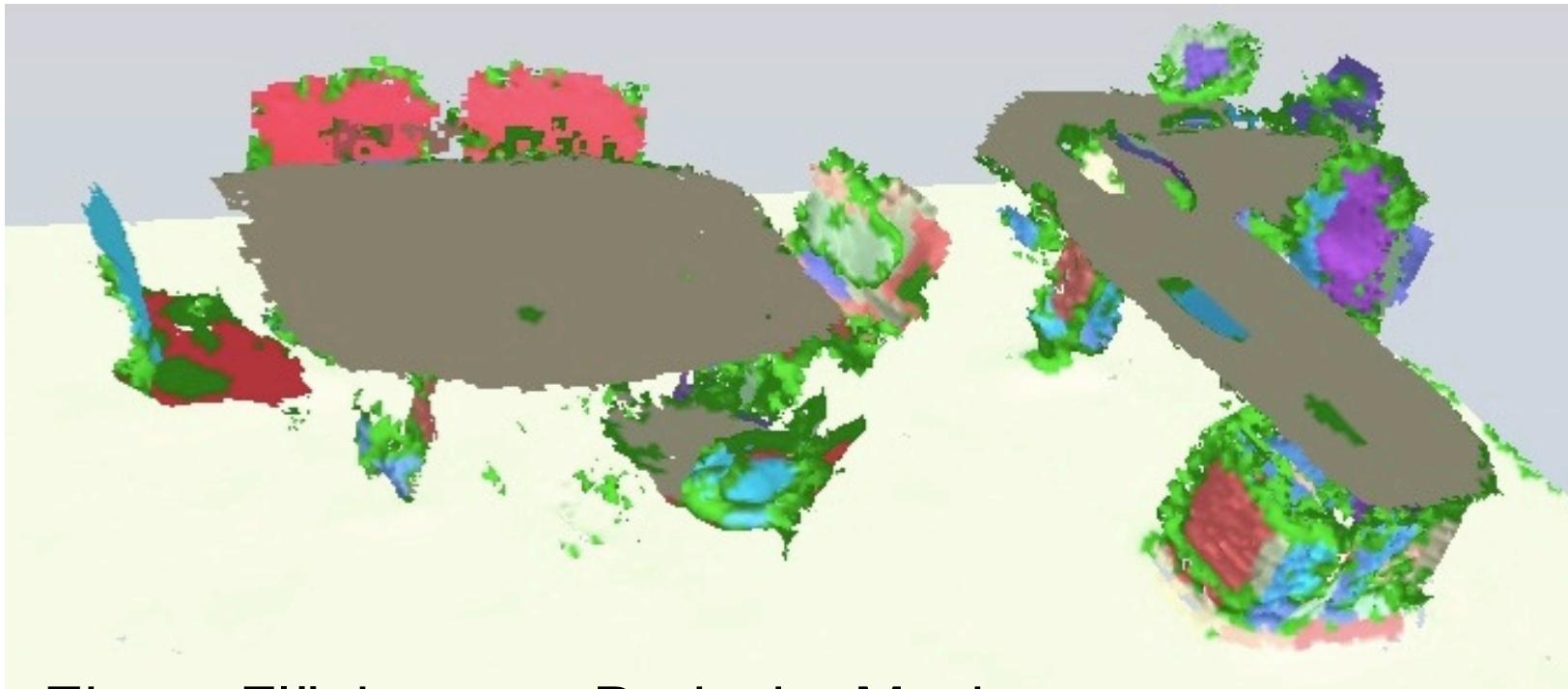
Modifikation (ignoriere Fehler aus Unterschied Scan/Sample)

- Bestimme Korrespondenz Modell/Daten gemäß belegten/leeren Voxel Bins

Beispiel: Bürodatensatz



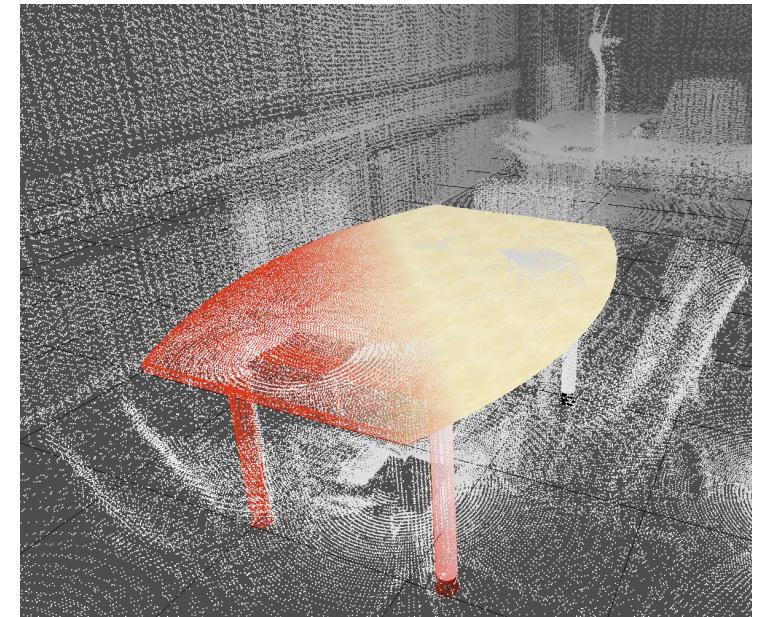
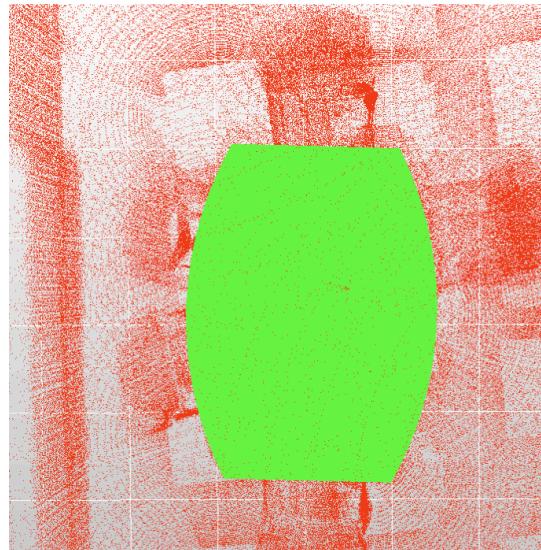
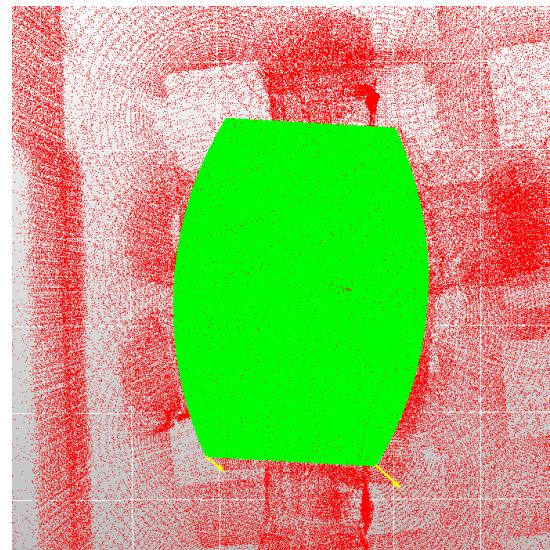
Beispiel: Primitive & Hypothesen



- Ebene Flächen aus Dreiecks-Mesh
(Benachbarte Dreiecke mit ähnlichen Normalen)
- Nicht-ebene Oberflächen in grün
- Tisch(platten)-Hypothesen in grau

Beispiel: Verifikation

Verifizierte Tisch-Objekt und -Pose
durch ICP-Matching des Punkt-
Samples an der hypothetischen Pose



Passende CAD
Tischoberfläche vor und
nach ICP-Matching

Gesamtergebnis



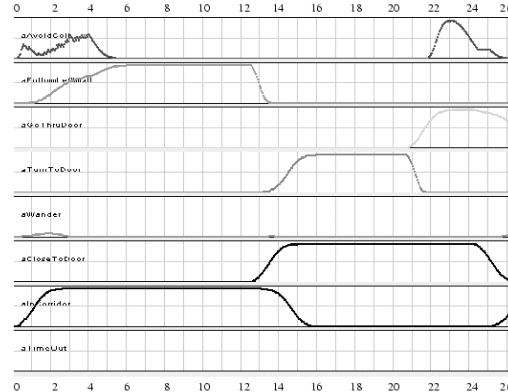
Zusammenfassung Umgebungsinterpretation

- Verankerung (*symbol grounding*, Objektverankerung) ist ein extrem schwieriges Problem
- Viele halten es auch für extrem wichtig
- Man muss es lösen, wenn man z.B. mit Robotern auf Basis der Sensordaten die Umgebung interpretieren will
- Identifikation von Objekten/Instanzen bildet den Einstieg in Symbolverankerung
- Es ist abhängig von der Art der Sensordaten, die vorliegen
- Semantische Karten geben einmal identifizierten Objekten Persistenz und erlauben, sie mit Techniken der Wissensverarbeitung zu behandeln
- Es gibt nur verstreute Literatur-Beispiele für den Aufbau semantischer Karten, und es gibt keine Methodik dafür

Kapitel 9: Roboterkontrollarchitekturen

1. Zum Einstieg: Worum geht es?
2. Sensorik
3. Sensordatenverarbeitung
4. Fortbewegung
5. Lokalisierung in Karten
6. Kartierung
7. Navigation
8. Umgebungsdateninterpretation
9. Roboterkontrollarchitekturen

Ausblick

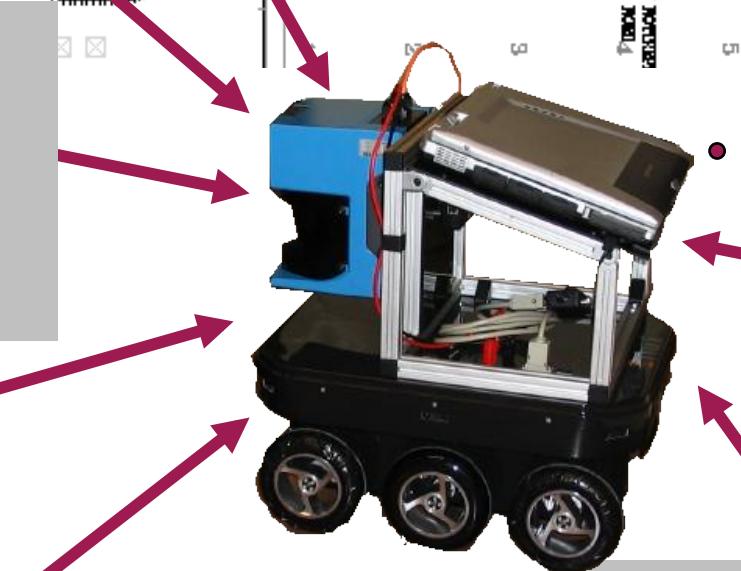
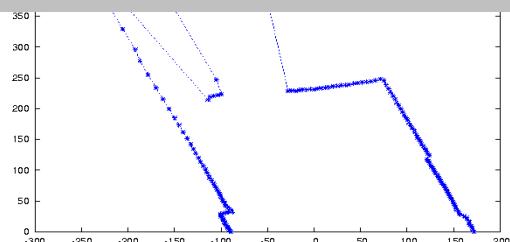


Wissen ist Last

$v_{L,set}?$ $v_{R,set}?$

Aktuelle Informationen

- Liftwartung 8:00–10:00
- Sekretariat in Urlaub



“Ewige Wahrheiten”

- Du sollst nie Deine Akkus leerfahren!
- Leitungs-Aufträge haben Priorität!

Tagesplan für 5.2.2007

- 5. Etage kartieren
- 10:00 Post austragen
- 13:30 Besucher von Lift abholen

Robotersteuerungsarchitektur

Robotersteuerungssoftware ist Software! Qualitätskriterien:

- **Nutzungssicht**: Klares Nutzungsdesign, Robustheit, ...
- **Codesicht**: saubere Modularisierung, ggf. Verwendung von Standards, transparenter Kontroll-/Datenfluss, Dokumentation, ...
- **Prozesssicht**: Systematische Verifikation/Test, Lebenszyklusmodell, ...

Inwiefern ist Robotersteuerungssoftware speziell?

- Für viele Detailprobleme keine Standardalgorithmen
- Anforderungen/Randbedingungen oft unklar/variabel
- Roboter sind eingebettete Systeme
(geschlossene Steuerung, Sensordatenströme, Echtzeitanforderungen)
- Daten unterschiedlicher Granularitäten und Zeitskalen
- Zykluszeiten sind auf allen Zeitskalen einzuhalten

**Unklar, wie dafür eine gute Softwarearchitektur aussieht!
Gute Architektur ist aber Schlüssel für gute Software!**

Roboterkontrollarchitekturen

Robot Control Architectures

- ... werden allgemein als wichtiges Thema angesehen
- ... sind als Forschungsgebiet „strategisch“ schwierig
 - „boxes-and-arrows“ Ergebnisse
 - Beitrag der Architektur zu Roboterperformanz nicht quantifizierbar – kein Fortschritt nachweisbar
 - Architektur i.d.R. nicht unabhängig von anderen Elementen (Sensorik, Basisalgorithmen, Fahrgeschwindigkeit, ...)
- ... werden oft verengt auf den Daten/Kontrollfluss beschrieben