Modellierung von Positionssensoren

Jörg Roth
Fachbereich Informatik
Fernuniversität Hagen

Position und Positionssensorik

- Die Position ist eine der wichtigsten Einflussgrößen für ortsbezogenen Dienste
- Im Gegensatz zu vielen anderen Größen wird die Position *gemessen* nicht *berechnet*:
  - Sensorik nicht ideal bezüglich Genauigkeit
  - Keine vollständige Abdeckung, Verfügbarkeit
- In vielen Projekten wird die Position verwendet wie eine algorithmisch ableitbare Größe:
  - Keine explizite Modellierung der "negativen" Eigenschaften
  - Resultate werden oft unter idealen Bedingungen abgeleitet (DGPS, 100% verfügbar)

Ziele

- Die Robotik hat große Erfahrungen in diesem Gebiet. Viele Randbedingungen der Robotik treffen jedoch nicht zu
  - Hardware-Kosten, Energieverbrauch
  - Viele Verfahren prinzipiell nicht einsetzbar, Fokus auf Indoor
  - Beliebige CPU-Ressourcen für die Sensordatenfusion

Ziele:

- Explizites Modell für Positionsmessungen und -sensoren
- Propagieren der Eigenschaften von Positionsangaben
- Kombinationen verschiedener Sensoriken

Das Nimbus-Positionsmodell

Nimbus (siehe u.a. Fachgespräch 04 in Hagen)

Einige Eigenschaften:

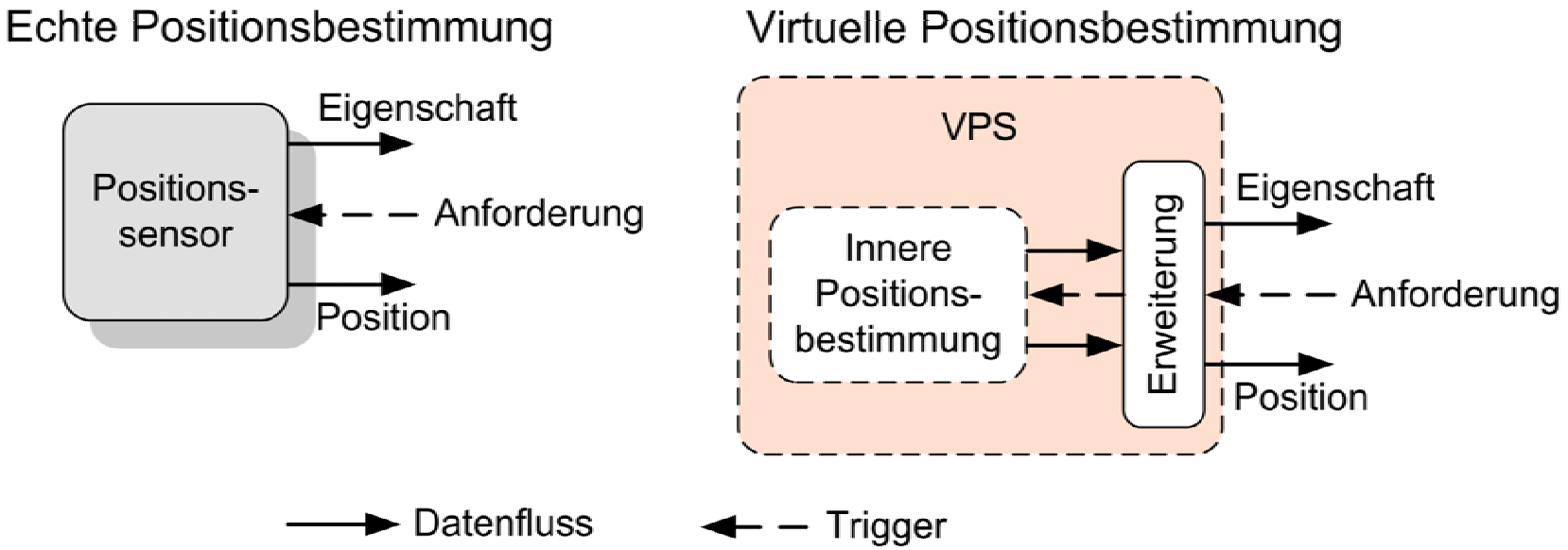
- Selbstorganisierende Infrastruktur
- Stellt *semantische Positionen* zur Verfügung
- Trigger-Services, Geocast

Für diesen Vortrag relevant:

- *Sensormodell* auf der Basis *virtueller Positionsbestimmungssysteme* (*VPS*)
- *Positionsmodell* auf der Basis von *Areas*
- Algorithmen zur *Erweiterung* von Positionsdaten

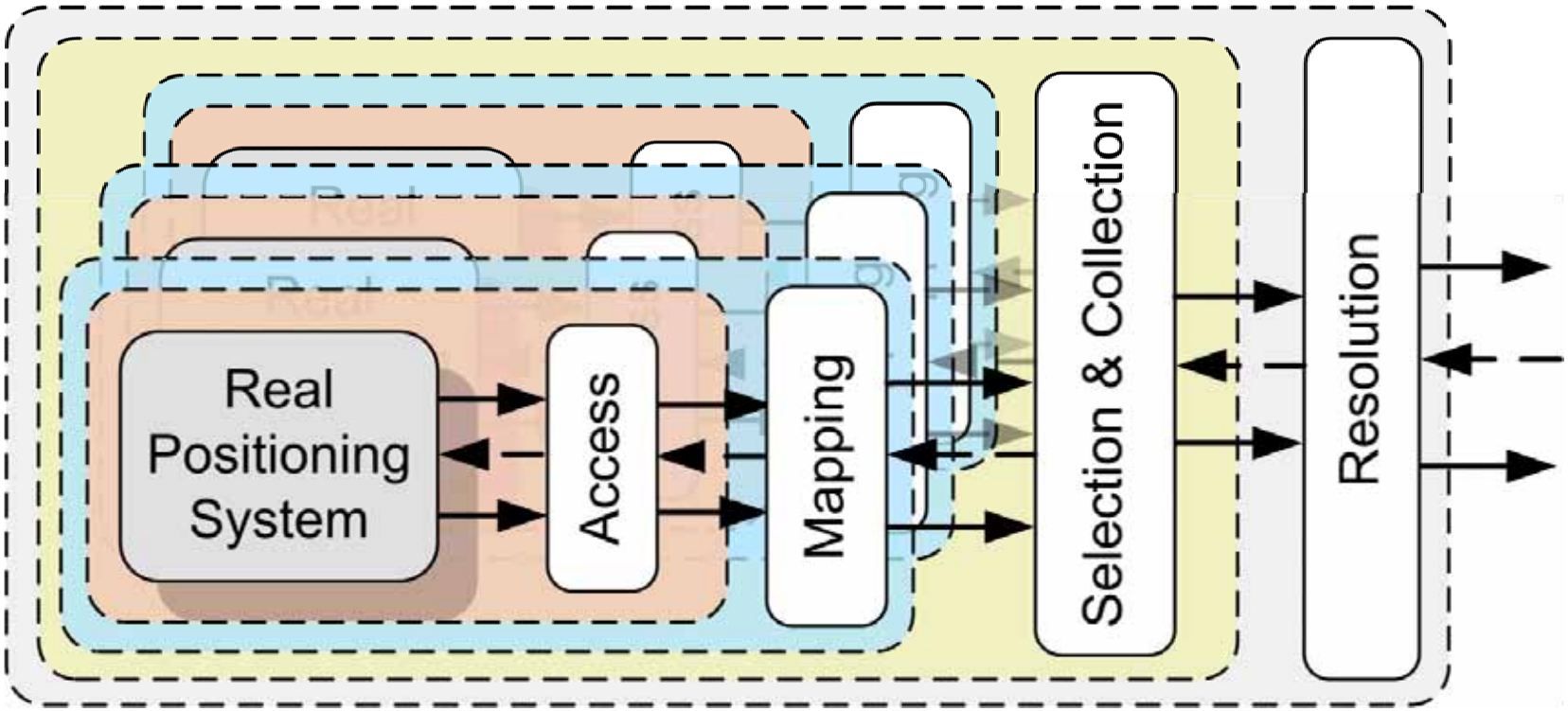
Virtuelle Positionsbestimmung

- Virtuelle Positionsbestimmungssysteme (VPS) dienen der Strukturierung des Problems
  - Verschiedene Stufen der Erweiterung
  - Definierte Schnittstellen



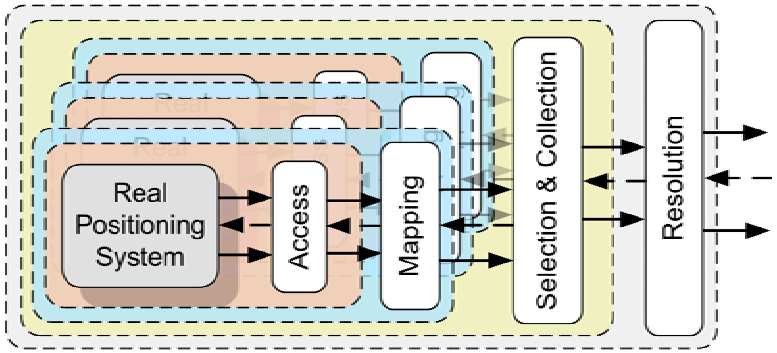
Die Anordnung der VPS

- Eine sinnvolle Anordnung:



Erweiterung der Sensordaten

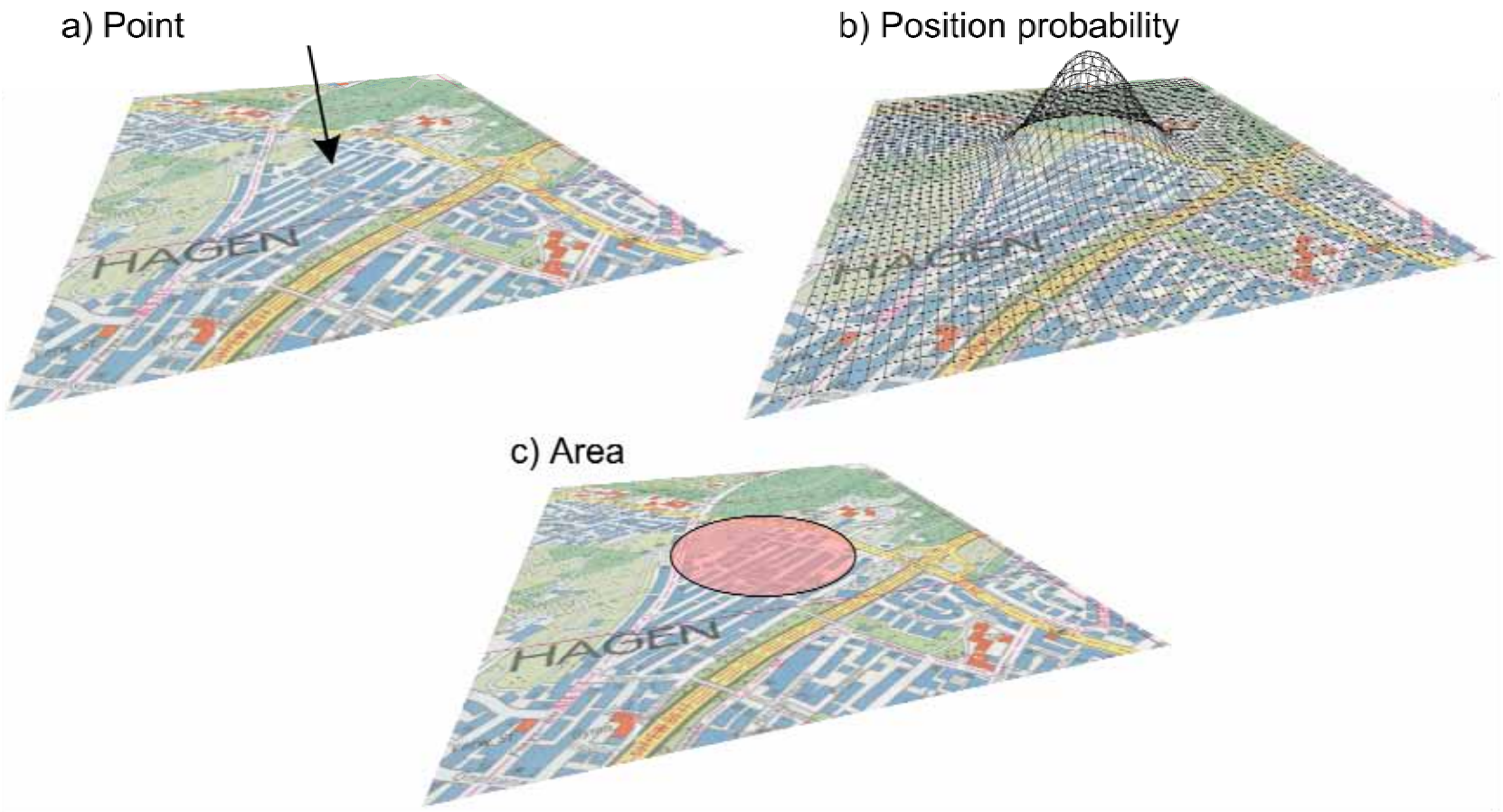
Typen der Erweiterung



- Access: Zugriff auf die Sensorik über *Treiber*
- *Mapping*: Umwandlung von Rohdaten in global eindeutige Koordinaten
  - Koordinatentransformation
  - Ggfs. mit Hilfe von *Mapping Servern*
- *Selection & Collection*: Auswahl von Systemen, die aktiviert werden und Fusion der gesammelten Sensordaten
- *Resolution*: Abbildung von physikalischen in semantische Positionen

Modellierung von Sensordaten

- 3 Möglichkeiten



Modellierung von Sensordaten

- *Points*: sehr starke Idealisierung
- *Probability*: schwierig, wenn man nicht nur normalverteilte Aufenthaltswahrscheinlichkeiten hat (z.B. COO-basiert, GSM-Zellen)
- Nimbus verwendet das *Area*-Modell
  - Normalverteilte und COO-basierte Positionsangaben können im gleichen Modell ausgedrückt werden
  - Algorithmen auch auf kleinen Endgeräten lauffähig
  - Informationsverlust tragbar
  - Sensoren mit unbekannter Verteilung modellierbar

Auswahl der Sensordaten

Selection & Collection:

- In der Robotik: Aktivierung *aller* verfügbaren Sensoren, Fusion aller Sensordaten
- Hier: Es sollten nur die *notwendigen* Positionssensoren aktiviert werden
  - Es entstehen u.U. *Kosten* für Positionsmessungen (AGPS, DGPS per SMS, Timing Advance nur bei Verbindung, Batterie)
  - Anwendung muss Bedarf und Kostenlimit ausdrücken (*Quality of Service* für Positionsdaten)
  - Problem: aktivierte Systeme müssen nicht unbedingt Positionsdaten bereitstellen (z.B. GPS in Gebäuden), dennoch Kosten durch Aktivierung

Selection

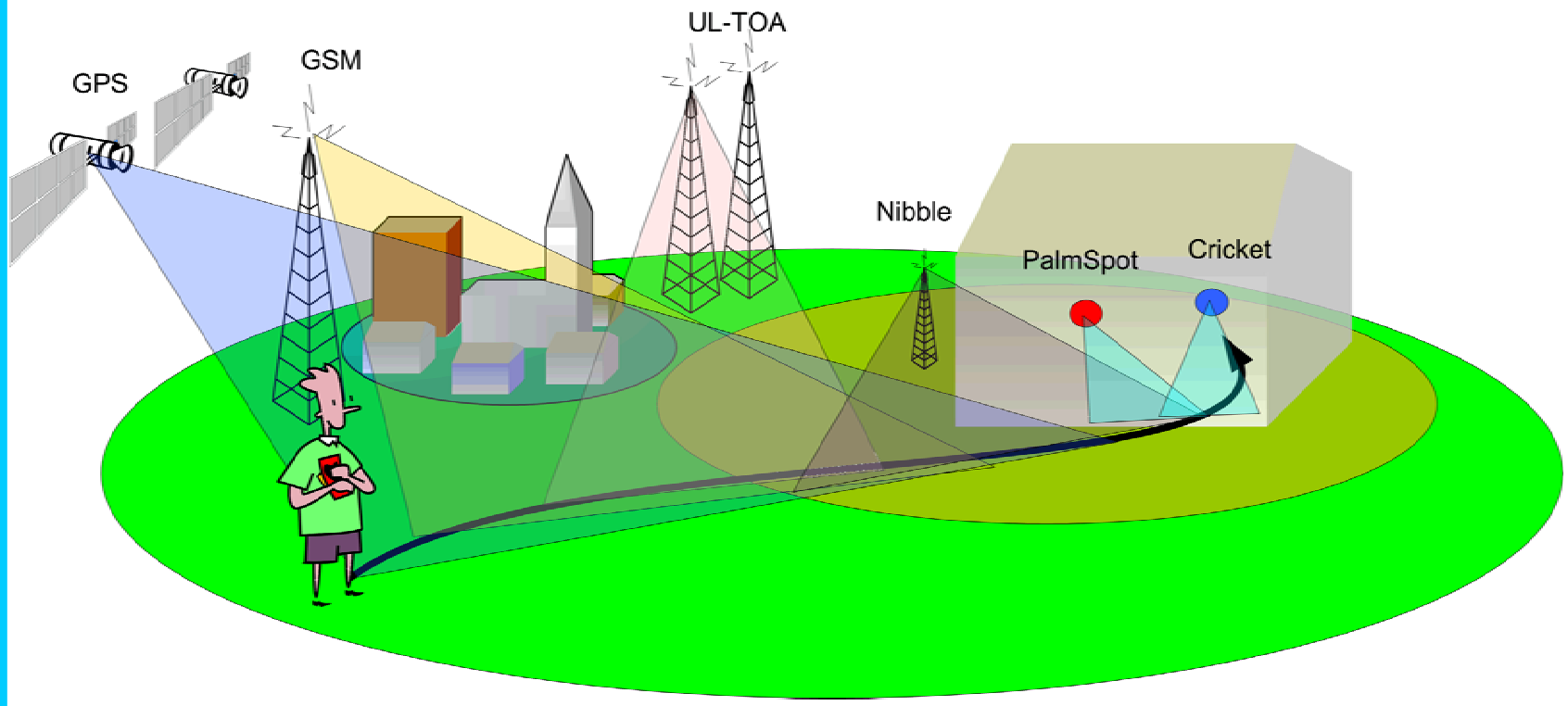
Der Nimbus Selection-Algorithmus:

- Aktiviere kurzfristig *alle* verfügbaren Systeme, messe dabei Verfügbarkeit, Kosten und Genauigkeit
- Ermittle eine Auswahl, so dass
  - das Kostenlimit nicht überschritten wird
  - die erreichte Genauigkeit maximal ist
- Aktiviere für eine bestimmte Zeit nur die *ausgewählten* Systeme
- Nach einer bestimmten Zeit oder wenn zu wenig verfügbar: beginne von vorne

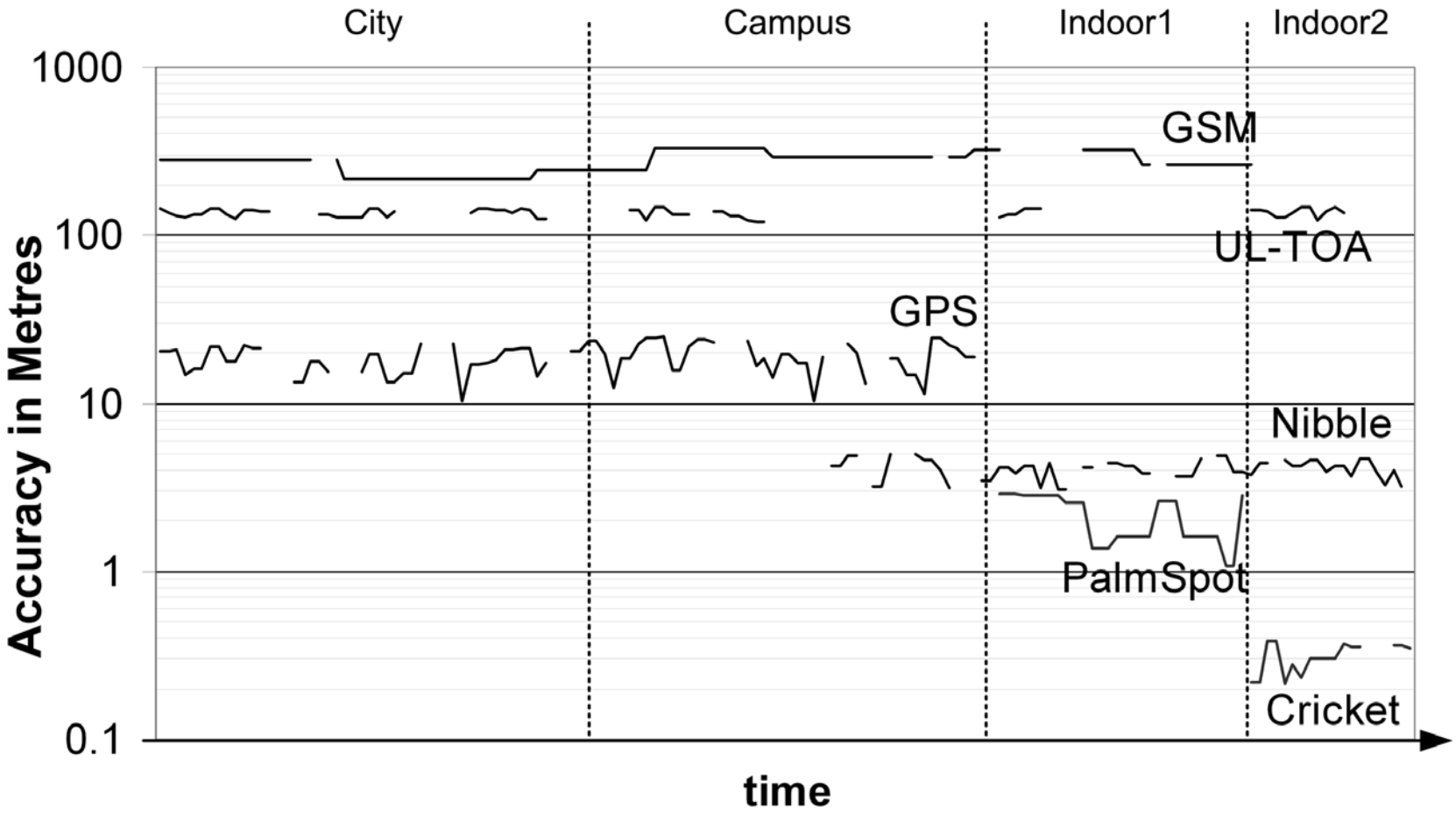
Selection

- Der Algorithmus lässt sich durch zwei Parameter *m* (maximale Kosten), *n* (innere Iterationen) steuern
  - Begrenzung der maximalen Kosten
  - Abschätzung für durchschnittliche Kosten
- Das Auswahlproblem lässt sich auf 0/1-Knapsack zurückführen
  - "Güter" sind die Positionsbestimmungssysteme
  - "Gewinn" ist die Genauigkeit
  - "Gewicht" sind die Kosten
  - Lösung ist unproblematisch, da kleine Mengen von "Gütern" verwendet werden (etwa max. 10 Stück)

Simulation



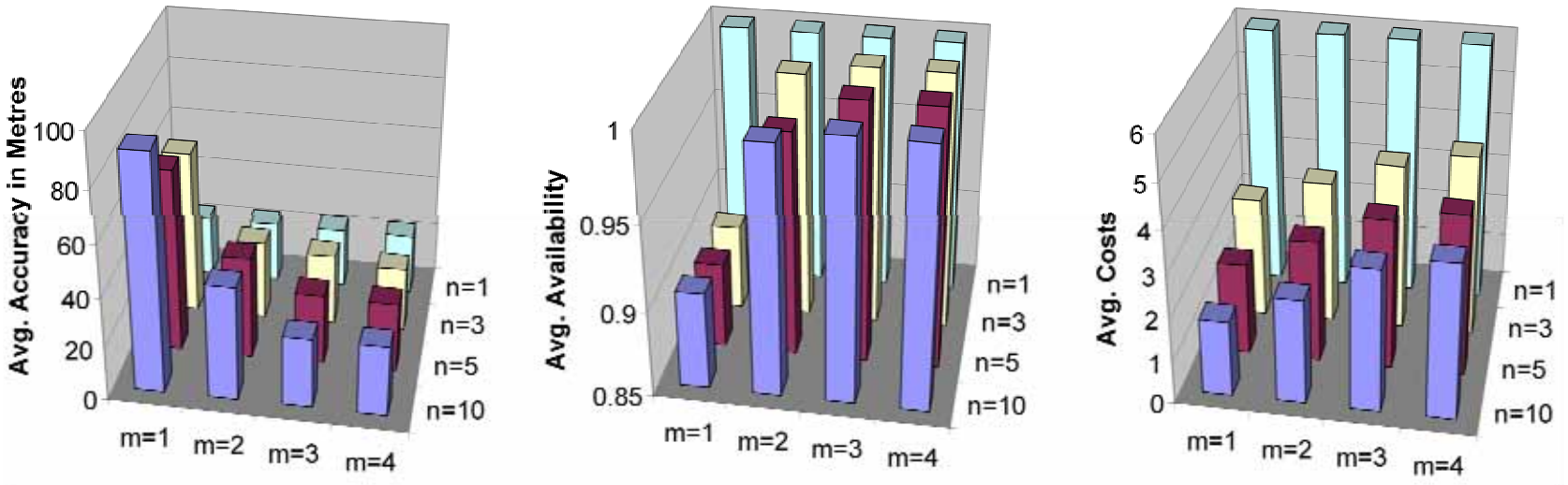
Beispiellauf



$m = 3, n = 10$

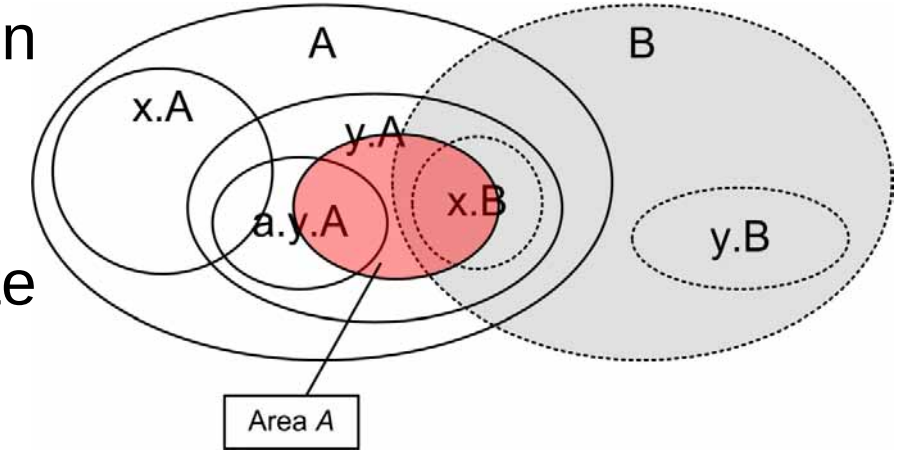
Simulationsergebnisse

- Algorithmus stellt sich schnell auf neue Umgebungsbedingungen ein
  - implizites Handover
- Kostenlimit wird berücksichtigt
- Steigerung der Verfügbarkeit und Genauigkeit



Weitere Eigenschaften

- *Collection*-Komponente:
  - Wird auf die geometrische Schnitt-Operation zurückgeführt
- *Resolution*-Komponente:
  - Im Gegensatz zum Point-Modell ist man nicht mehr eindeutig innerhalb oder außerhalb einer semantischen Position
  - Effizienter Algorithmus existiert, der auf der Punkt-Variante aufbaut



Zusammenfassung und zukünftige Arbeiten

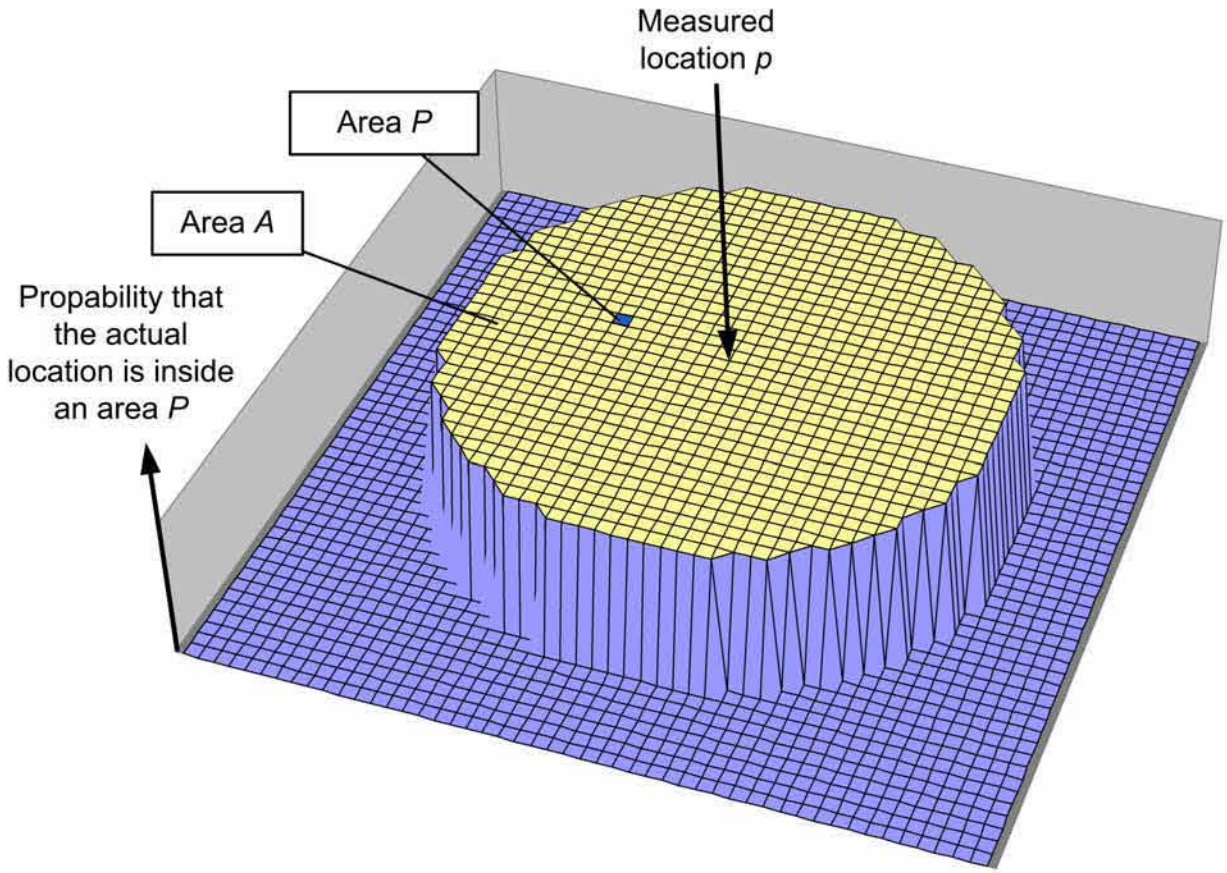
- VPS:
  - Strukturierung des Problems
  - Sinnvolle Software-Architektur
  - Entkopplung der Komponenten
- Area-Positionsmodell:
  - Effiziente Realisierung für Algorithmen existiert
- Kosten und QoS
  - Selection & Collection
- Zukünftige Arbeiten: bessere Modellierung der Positionsdaten
  - Ideal: Area und Probability gemischt

Joerg.Roth@Fernuni-hagen.de

http://dreamteam.fernuni-hagen.de

Modellierung von Sensordaten

Areas sind spezialfall von Wahrscheinlichkeiten



Simulation

Vorteile von VPSen: Simulation (Achtung, nicht mit der anderen Sim im Vortrag verwechseln)