Jia Lei Du

Zellenbasierte Dienst-Entdeckung für Roboternetzwerke

Bibliografische Information Der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen National-bibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar

Satz und Gestaltung: Jia Lei Du

Hersteller: Verlagshaus Monsenstein und Vannerdat OHG
Druck · Buch · Verlag
Münster

Printed in Germany

Geleitwort

Verbindendes Forschungsziel der von mir geleiteten Fachgruppe Schaltungstechnik ist der systematische Entwurf und der bedarfsgerechte Einsatz von mikroelektronischen Systemen in konkreten Anwendungen der Informations- und Automatisierungstechnik. Unsere Aktivitäten umfassen Arbeiten auf System- und Schaltkreisebene sowohl in digitaler als auch analoger Schaltungstechnik. Besondere Berücksichtigung finden massiv-parallele Realisierungsvarianten sowie die Bewertung der Ressourceneffizienz entsprechender Implementierungen. Ressourceneffizienz bedeutet hier, mit den physikalischen Größen Raum, Zeit und Energie sorgfältig umzugehen.

Die zunehmende Verfügbarkeit von kostengünstigen mobilen Roboterplattformen hat in den letzten zehn Jahren zu einem verstärkten Forschungsinteresse an Multi-Roboter-Systemen geführt, die eine größere Robustheit und eine höhere Effizienz bei der Lösung von komplexen Aufgabenstellungen ermöglichen. Der Entwurf, die Implementierung und die Evaluation von kooperierenden Robotergruppen beinhalten eine Vielzahl von neuen wissenschaftlichen und technischen Herausforderungen. Eine davon ist die Erforschung von geeigneten Kooperationsmodellen. Die aus der Informatik bekannten Multiagentensysteme können für Multi-Roboter-Systeme nicht direkt angewandt werden, da letztere als physikalische Objekte in physikalischen Umgebungen agieren müssen.

Herr Jia Lei Du beschäftigt sich in seiner Arbeit mit der Frage, wie in einem Netzwerk aus heterogenen, lose gekoppelten Robotern effizient diejenigen Roboter lokalisiert werden können, die einen gewünschten Dienst anbieten. Zur Lösung dieser Aufgabenstellung greift er auf Konzepte aus dem Bereich der servicebasierten Software-Architekturen zurück. Servicebasierte Architekturen haben das Potential, die Entwicklung und die Nutzung großer Multi-Roboter-Systeme zu vereinfachen. Die effiziente Dienst-Entdeckung ist eine grundlegende Komponente in servicebasierten Systemen.

Im Rahmen seiner wissenschaftlichen Arbeiten hat Herr Jia Lei Du ein neues Protokoll zur zellenbasierten Dienst-Entdeckung (CSD) entwickelt, mit Hilfe von Simulationen analysiert und prototypisch auf Minirobotern realisiert. CSD basiert auf einer gitterartigen Zellenstruktur des Operationsgebietes der Roboter mit jeweils einem Hauptknoten pro Zelle, der die Dienste anderer Roboter (Knoten) der Zelle verwaltet. Durch die Einführung einer Zellenstruktur wird der Einfluss von Knotenbewegungen auf die Netzwerkverwaltung reduziert. Wird ein Dienst nicht in einer Zelle gefunden, werden Hauptknoten benachbarter Zellen befragt. Als wichtige Kenngrößen für die Beurteilung der Leistungsfähigkeit werden die Anzahl der Zellenwechsel der Roboter (Knoten), die Quote der erfolgreichen Dienst-Entdeckungen, das Nachrichtenaufkommen sowie die Latenz zwischen Suchanfrage und Suchergebnis herangezogen. Die Simulationsergebnisse bestätigen seine formalen Schlussfolgerungen, dass CSD eine skalierbare und gute Lösung für Roboternetzwerke mit mittlerer bis hoher Mobilität der Netzwerkteilnehmer ist.

Herr Jia Lei Du hat seine Dissertation im Graduiertenkolleg „Automatische Konfigurierung in offenen Systemen“ durchgeführt. Der Deutschen Forschungsgemeinschaft sei an dieser Stelle noch einmal ausdrücklich für die finanzielle sowie strukturelle Unterstützung und das damit verbundene Vertrauen gedankt.

Prof. Dr.-Ing. Ulrich Rückert

UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

FAKULTÄT FÜR ELEKTROTECHNIK, INFORMATIK UND MATHEMATIK

Zellenbasierte Dienst-Entdeckung für Roboternetzwerke

Zur Erlangung des akademischen Grades

DOKTORINGENIEUR (Dr.-Ing.)

der Fakultät für Elektrotechnik, Informatik und Mathematik
der Universität Paderborn
vorgelegte Dissertation
von

Dipl.-Ing. Jia Lei Du
Shaoxing

Referent: Prof. Dr.-Ing. Ulrich Rückert
Korreferent: Prof. Dr. Friedhelm Meyer auf der Heide

Tag der mündlichen Prüfung: 30.05.2007

Paderborn, den 30.05.2007

Diss. EIM-E/232

Zusammenfassung

Wenn in Zukunft mobile autonome Roboter in großer Zahl in unserer täglichen Umwelt installiert werden, wird eine Zusammenarbeit zwischen ihnen für einen erfolgreichen Betrieb notwendig sein. Ein möglicher Ansatz für die Organisation einer großen Anzahl von Robotern ist die Spezialisierung der Roboter auf bestimmte Funktionalitäten und die Bereitstellung und Inanspruchnahme dieser Funktionalitäten in Form von Dienstleistungen. Eine Grundaufgabe hierbei ist es, geeignete Roboter aus der Umgebung zu lokalisieren, die die gewünschten Dienste erbringen können. In dieser Arbeit wird das Cell-based Service Discovery-Protokoll (CSD) für die Dienst-Entdeckung in Roboternetzwerken entwickelt. Die Grundidee basiert auf der Bildung einer Zellstruktur unter Nutzung der Positionsinformationen der Roboter und der Wahl von Hauptknoten in den Zellen, die für die Verwaltung der gewöhnlichen Knoten und ihrer Dienste verantwortlich sind. Die Lösung wird analytisch und in Simulation untersucht und mit anderen Lösungen verglichen. CSD entfaltet seine Stärken in Szenarien, in denen eine skalierbare Lösung für Netzwerke mit mittlerer bis hoher Mobilität der Knoten benötigt wird. Abschließend wird eine vereinfachte Version des Protokolls auf einem realen Multi-Roboter-System implementiert und anhand eines Beispielszenarios werden die Vorteile servicebasierter Multi-Roboter-Systeme demonstriert.

Abstract

If robots are deployed in large numbers in our environment in future, collaboration between them may be essential for a successful operation. One potential approach for the organisation of a large number of robots is their specialisation in certain functionalities and the provision and use of these functionalities as services. One crucial task in this approach is the localization of robots in the environment that can provide the looked-for services. In this thesis the Cell-based Service Discovery (CSD) protocol is developed. The basic idea is to form a cell-based grid using the position information of the robots. Master nodes in each cell are responsible for the management of ordinary nodes and their services. The solution is analyzed analytically and in simulation, and compared to other solutions. CSD has its strength in scenarios where a scalable solution for networks with average to high node mobility is required. A simplified version of the protocol is implemented on a real multi-robot system and using an example scenario the advantages of service-based multi-robot systems are demonstrated.

Danksagung

Zuallererst gilt mein Dank Herrn Professor Ulrich Rückert, ohne ihn wäre diese Arbeit nicht entstanden. Er hatte das Vertrauen, mir freie Hand bei der Wahl meines Dissertationsthemas zu lassen und mich dann auf diesem Weg zu unterstützen. Neben der fachlichen Unterstützung bei der Erstellung meiner Arbeit habe ich während meiner Zeit als Mitarbeiter in Professor Rückerts Fachgruppe zugleich viel in menschlicher Hinsicht von ihm gelernt. Ich schätze und respektiere ihn als Vorgesetzten und Persönlichkeit für seine Fähigkeit, Menschen mit Wohlwollen, Vertrauen und Respekt zu führen. Ich möchte mich ebenfalls herzlich bei Herrn Professor Meyer auf der Heide für die Übernahme des Korreferats bedanken.

Die Arbeit am Heinz Nixdorf Institut war nicht zuletzt aufgrund der hervorragenden Arbeitsatmosphäre sowohl erfolgreich wie auch angenehm. Hierfür bedanke ich mich bei meinen Kollegen der Fachgruppe Schaltungstechnik, insbesondere den Kollegen aus der Robotikgruppe unter Leitung von Dr.-Ing. Ulf Witkowski.
Weiterhin möchte ich mich bei meinen Bürokollegen der ersten und zweiten Runde bedanken, die mich in ihre Mitte aufgenommen haben. Ohne sie hätte das Arbeiten nur halb soviel Spaß gemacht.
Mein Dank gilt ebenfalls den Kollegen aus der Mittagsrunde, die mich stets mit interessanten Diskussionen und unterhaltsamen Gesprächen aus meinem Mittagstief gerissen haben.

Mein vorletzter Dank geht an die Deutsche Bahn AG. Wenn ich nicht damit beschäftigt war, Beschwerdebriefe an sie zu schreiben, habe ich während der Zugfahrten ohne störende Unterbrechung durch Emails und Telefon an meiner Dissertation gearbeitet.

Diese Arbeit ist im Rahmen des Graduiertenkollegs „Automatische Konfigurierung in offenen Systemen“ entstanden. Hierfür möchte ich mich abschließend herzlich bei der Deutschen Forschungsgemeinschaft für ihre Unterstützung bedanken.

Inhaltsverzeichnis

1. Einleitung

Den Versprechen von Visionären zum Trotz sind Roboter noch immer nicht die intelligenten, autonomen Wesen, die immer wieder vorhergesagt wurden. Auch heute sind die meisten im professionellen Bereich eingesetzten Roboter nach wie vor immobile Industrieroboter, die für strukturell einfache, monotone Aufgaben eingesetzt werden [UNEC04]. Die aus Fernsehbildern bekannten mobilen Roboter der *NASA (National Aeronautic and Space Administration)*, wie beispielsweise die Marsroboter *Spirit* und *Opportunity*, sind dagegen teure Einzelanfertigungen, die weitgehend ferngesteuert werden (Abb. 1-1) [CAM+03]. Die Vision von intelligenten, mobilen Robotern, die als allseits verfügbare Helfer in unserer täglichen Umwelt neben und mit uns leben, scheint noch weit von ihrer Erfüllung zu sein. Doch ebenso waren angesichts der rasanten technologischen Entwicklung der letzten Jahrzehnte in den für die Robotik relevanten Bereichen wie der Elektrotechnik, dem Maschinenbau und der Informatik und dank der zunehmenden Verschmelzung und Integration der wissenschaftlichen Disziplinen die Chancen niemals höher als jetzt, diesem Ziel tatsächlich nahe zu kommen. Dies zeigt sich auch an der steigenden Anzahl von – bisher noch sehr einfachen – mobilen Haushaltsrobotern, die weltweit zunehmende Verbreitung finden.

Der große Vorteil mobiler Roboter liegt in ihrer Fähigkeit der flexiblen, nicht ortsgebundenen, physikalischen Manipulation ihrer Umwelt. Durch den technischen Fortschritt in der Computertechnik haben wir als Anwender bereits erste Erfahrungen mit quasi intelligenten Maschinen gemacht. Um jedoch den Vorteil mobiler Roboter pointiert darzustellen: Kein noch so leistungsfähiger Computer wird jemals in der Lage sein, uns ein Glas Saft zu bringen. Diese Fähigkeit der Mobilität und physikalischen Manipulation der Umwelt ist ein echter Mehrwert und nicht durch Steigerungen anderer Leistungsmerkmale, wie beispielsweise der Rechen- oder Speicherkapazität kompensierbar. Durch die Ausstattung mit Sensoren und Aktoren können sich mobile Roboter in der realen Welt bewegen und diese manipulieren, was einen Fortschritt gegenüber derzeitigen Computersystemen und Industrierobotern darstellt. Sie können jedoch ebenso direkt mit Computern auf binärer Ebene kommunizieren und interagieren und die virtuelle Welt ohne eine dazwischen liegende Repräsentationsschicht betrachten und beeinflussen. Damit haben mobile Roboter die einzigartige Fähigkeit, in beiden Welten, der digitalen und der realen Welt, natürlich navigieren und agieren zu können,

im Unterschied und im Vorteil zu den heutigen technischen Systemen und auch uns Menschen.

Wenn mobile autonome Roboter in großer Zahl in unserer täglichen Umwelt installiert werden, wird eine Zusammenarbeit zwischen ihnen für einen erfolgreichen Betrieb notwendig sein. Es ist weder arbeitsorganisatorisch sinnvoll noch technisch möglich, alle Funktionen in jeden Roboter zu integrieren. Weitere Vorteile der Arbeitsteilung sind neben der parallelen Bearbeitung von Aufgaben auch das Potential, die Robustheit und Zuverlässigkeit des Gesamtsystems zu erhöhen. So kann ein Multi-Roboter-System auch im Falle des Versagens einzelner Roboter möglicherweise die geforderte Aufgabe trotzdem erfüllen, sei es durch Redundanz in der Anzahl der Roboter oder durch Überschneidungen in den Aufgabentypen, die die Roboter des Systems bearbeiten können. Die Zuverlässigkeit des Gesamtsystems wird auch dadurch erhöht, dass die einzelnen Roboter eines Multi-Roboter-Systems weniger komplex sein können als ein Einzel-Roboter-System, in dem alle benötigten Funktionen integriert sind. Schließlich kann durch dynamische Rekombination der Roboter eine erhöhte Flexibilität beim Lösen der Aufgaben und eine effizientere Bearbeitung der Aufgaben erreicht werden.
Auf der anderen Seite kann sich bei Multi-Roboter-Systemen die Entwurfskomplexität erhöhen. Ursprüngliche lokale Entscheidungsprozesse und Handlungen erfordern nun Kommunikation und Kooperation. Informationen und Entscheidungsprozesse sind verteilt und es werden zusätzliche Fehlerquellen wie die Kommunikationsinfrastruktur eingeführt.

Letztendlich wird die Beherrschung von großen, komplexen Multi-Roboter-Systemen davon abhängen, inwieweit die Fähigkeit der Selbstorganisation in das System integriert werden kann. Ein möglicher Ansatz hierfür ist – ähnlich der Organisation moderner Wirtschaftssysteme – die Spezialisierung der Roboter auf bestimmte Fähigkeiten und die Bereitstellung und Inanspruchnahme dieser Fähigkeiten in Form von Dienstleistungen, um Aufgaben zu lösen. Die Zusammenarbeit in großen Multi-Roboter-Systemen wird ad hoc durch immer wieder neue, flexible Kombination der Roboter mit ihren unterschiedlichen Fähigkeiten erfolgen. Ebenso können sich Roboter schon im Voraus zu Gruppen zusammenschließen und als Gruppe einen Dienst anbieten. Hier ist eine beliebige Hierarchisierung möglich. Eine Grundaufgabe bei diesem Ansatz zur Organisation von Multi-Roboter-Systemen ist es, geeignete Roboter aus der

Umgebung zu lokalisieren, die die zu bearbeitenden Aufgaben erfüllen können. Genau mit dieser Fragestellung setzt sich diese Arbeit auseinander. Wie können in einem Netzwerk aus mobilen Robotersystemen effizient Roboter lokalisiert werden, die die gewünschten Dienste erbringen?

Ziel dieser Arbeit ist die Entwicklung eines Protokolls für die Dienst-Entdeckung in Roboternetzwerken. Roboternetzwerke sind charakterisiert durch eine hohe Mobilität der Teilnehmer, die Verfügbarkeit von Positionsinformationen und ein hohes Verhältnis von Dienstbekanntmachungen zu Dienstsuchen. Die ersten beiden Eigenschaften ergeben sich aus der originären Aufgabe mobiler Roboter: die Ausführung physischer, örtlich verteilter Aufträge. Die Netzwerkteilnehmer werden daher einerseits regelmäßig in Bewegung sein, im Unterschied zum Beispiel zu Sensornetzwerken. Damit einher geht eine hohe Dynamik der Netzwerkstruktur. Andererseits können die Positionsinformationen der Roboter genutzt werden, um Effizienz und Skalierbarkeit des Protokolls zu erhöhen. Die physische Interaktion der Roboter bewirkt weiterhin eine reduzierte Rate der Dienstanfragen, da die physische Bearbeitung von Aufgaben, im Unterschied beispielsweise zu digitalen Interaktionen in Computernetzwerken, in der Regel Größenordnungen länger dauert. Hieraus ergibt sich wiederum in Kombination mit der Mobilität der Roboter ein hohes Verhältnis von Dienstbekanntmachungen zu Dienstsuchen. Weitere Eigenschaften, die allerdings allen mobilen Ad-hoc-Netzwerken gemeinsam sind, sind das unter Umständen völlige Fehlen von Infrastruktur und die drahtlose Kommunikation über Multi-Hop-Verbindungen.

In Kapitel 2 werden einige Grundlagen und existierende Lösungen für die Dienst-Entdeckung in mobilen Ad-hoc-Netzwerken vorgestellt. Der Hauptteil der Arbeit beginnt in Kapitel 3 mit einer Einführung in die Grundidee der servicebasierten Multi-Roboter-Systeme. Es werden mögliche Dienstleistungen in der Robotik betrachtet und ein Vergleich des servicebasierten Ansatzes mit anderen Entwurfsstrategien wie Schwarmintelligenz und Roboter-Teams durchgeführt. In Kapitel 4 wird das in dieser Arbeit entwickelte *Cell-based Service Discovery-Protokoll (CSD)* für die Dienst-Entdeckung in Roboternetzwerken vorgestellt. Es basiert auf der Bildung einer Zellstruktur unter Nutzung der Positionsinformationen der Roboter und der Wahl von Hauptknoten in den Zellen, die für die Verwaltung der gewöhnlichen Knoten und ihrer Dienste verantwortlich sind. Im Anschluss werden verschiedene Erweiterungen wie das

Suchen in mehreren Suchschritten eingeführt, um den Nachrichtenaufwand weiter zu reduzieren. Im zweiten Teil von Kapitel 4 und Kapitel 5 wird CSD analytisch und in Simulation untersucht und mit anderen Lösungen verglichen. Wie man sehen wird, entfaltet CSD seine Stärken in Szenarien, in denen eine skalierbare Lösung für Netzwerke mit mittlerer bis hoher Mobilität der Knoten benötigt wird. Zum Abschluss des Hauptteils wird in Kapitel 6 eine vereinfachte Version des Protokolls auf einem realen Multi-Roboter-System implementiert und anhand eines Beispielszenarios werden die Vorteile servicebasierter Multi-Roboter-Systeme demonstriert.

Abbildung 1-1. Illustration eines Marsroboters der NASA. Noch sind mobile Roboter oft Einzel-Roboter-Systeme. Der Austausch von Diensten ist ein Ansatz zur Beherrschung zukünftiger Multi-Roboter-Systeme. In dieser Arbeit wird ein Protokoll für die effiziente Entdeckung der in einem Roboternetzwerk verfügbaren Dienste entwickelt. (Bild: [NASA])

2. Grundlagen und Stand der Technik

In diesem Kapitel werden relevante Grundlagen für diese Arbeit und verwandte Forschungsarbeiten vorgestellt. Zuerst wird das positionsbasierte Routing-Protokoll *Greedy Perimeter Stateless Routing* erläutert, das in dieser Arbeit für das Routing der Nachrichten im Netzwerk verwendet wird. Anschließend werden existierende industrielle Lösungen für die Dienst-Entdeckung betrachtet und aufgezeigt, weshalb sich diese nur unzureichend für die Dienst-Entdeckung in Roboternetzwerken eignen. Danach wird auf aktuelle Protokolle aus der Forschung für die Dienst-Entdeckung in mobilen Ad-hoc-Netzwerken eingegangen. Abschließend werden einige bisher realisierte Roboternetzwerke und Beispiele für die Nutzung der Dienst-Entdeckung in Multi-Roboter-Systeme präsentiert.

2.1 Positionsbasiertes Routing: Greedy Perimeter Stateless Routing

Die Roboter werden zur Kommunikation so genannte mobile Ad-hoc-Netzwerke (MANETs) bilden. MANETs sind drahtlose Netzwerke aus mobilen Knoten, die durch ihre dezentralisierte Organisation und die potentiell hohe Dynamik der Netzwerkstruktur charakterisiert sind. Kommunikation über größere Distanzen in derartigen Netzwerken, die völlig ohne Infrastruktur funktionieren können, erfolgt über Multi-Hop-Verbindungen. Dabei fungieren Knoten auf dem Weg zwischen Quelle und Zielknoten als Vermittler (oder auch Router) und leiten erhaltene Nachrichten in Richtung ihrer Ziele weiter.

Routing-Algorithmen für mobile Ad-hoc-Netzwerke können in unterschiedliche Ansätze unterteilt werden. *Dynamic Source Routing* [JD96] ist ein typisches Beispiel für einen reaktiven Algorithmus. Die Route, die von einem Datenpaket genommen wird, wird erst bei Bedarf bestimmt. Der Quellknoten versucht, den Zielknoten innerhalb des Netzwerks durch Fluten einer Routenanfrage zu lokalisieren. Wenn der Zielknoten diese Routenanfrage erhält, schickt er eine Antwort an den Quellknoten und stellt so einen Verbindungsweg her. Im Gegensatz dazu werden Routen bei pro-aktiven Protokollen wie dem *Destination-Sequenced Distance-Vector Routing* [PB04] im Voraus aufgebaut

und gewartet, auch wenn zu dem Zeitpunkt keine Datenpakete verschickt werden. Hybride Protokolle wie das *Zone Routing Protocol* [HP02] verwenden pro-aktive Algorithmen für die Nachbarschaft eines Knotens und reaktive Protokolle für das Routing über größere Distanzen.

Positionsbasierte Routing-Protokolle nutzen wiederum die Positionsinformationen der Netzwerkknoten für das Routing [GSB04]. Eine Nachricht wird dabei an denjenigen Knoten innerhalb der Sendereichweite weitergegeben, der den geografischen Abstand zum Zielknoten minimiert. Positionsbasierte Protokolle gelten als skalierbar und effizient, weisen jedoch zwei Nachteile auf. Einerseits müssen alle Knoten über ihre eigenen Positionsdaten verfügen. Andererseits muss die geografische Position des Zielknotens erst einmal bekannt sein, bevor eine Nachricht an diesen geschickt werden kann. Es wurde dennoch ein positionsbasiertes Routing-Protokoll als Grundlage für die in dieser Arbeit vorgestellte Lösung zur Dienst-Entdeckung gewählt, weil sich diese beiden Nachteile im hier betrachteten Fall nicht auswirken: Denn Roboter navigieren in der realen Welt, damit benötigen sie ohnehin Positionsinformationen. Ebenso werden Roboter ihre Dienste zum Teil physikalisch austauschen, was die Kenntnis der relativen Position der involvierten Roboter und damit ein gemeinsames Koordinatensystem erfordert. Die Notwendigkeit von Positionsdaten ist somit keine wesentliche Einschränkung. Bezüglich der zweiten Anforderung ist es in diesem Fall so, dass derjenige Knoten, der den gewünschten Dienst im Netzwerk anbietet, ohnehin erst einmal lokalisiert werden muss. Wenn dieser Knoten gefunden wurde, kann er für den Aufbau der Kommunikationsverbindung gleich seine Positionsdaten mitschicken.

Im hier vorliegenden Fall können also die Vorteile positionsbasierter Routing-Algorithmen zur Geltung kommen, ohne dass sich die Nachteile dieser Lösungen auswirken. Aus diesem Grund wird in dieser Arbeit das positionsbasierte *Greedy Perimeter Stateless Routing (GPSR)* [KK00] als zugrunde liegendes Routing-Protokoll verwendet.

Das Grundprinzip von GPSR basiert auf zwei Modi. Es gibt einem so genannten *Greedy*-Modus, bei dem ein Paket an denjenigen Knoten innerhalb der Sendereichweite weitergegeben wird, der die Distanz zum Zielknoten minimiert. Für die Weiterleitung werden in diesem Fall nur lokale Informationen über die Positionen der benachbarten

Knoten benötigt. Es kann jedoch passieren, dass dieses Verfahren versagt, wenn das Paket in einem lokalen Minimum ankommt (Knoten x in Abb. 2-2), in dem kein Knoten innerhalb der Sendereichweite näher ist als der aktuelle, ohne dass jedoch der Zielknoten im nächsten Schritt kontaktiert werden könnte. In diesem Fall wird ein so genannter Perimeter-Modus aktiviert, der das Paket um dieses lokale Minimum herumleitet und dafür auch zeitweise akzeptieren kann, dass die Entfernung zum Zielknoten wieder zunimmt.

Die Voraussetzungen für GPSR sind, dass alle Knoten ihre Position in einem globalen Koordinatensystem kennen, dass alle Funkverbindungen bidirektional sind, dass die Knoten sich in etwa in einer Ebene befinden, und dass sendende Knoten (Quellen) die Positionen der Zielknoten (Senken) kennen. Außer den Positionen ihrer Nachbarn brauchen die Knoten fast keine Zustandsinformationen für die Weiterleitung der Pakete (daher das Attribut „stateless“). Durch die Verwendung ausschließlich lokaler Information ist GPSR skalierbar und auch bei ständigen Topologie-Änderungen können Routen schnell lokal korrigiert werden.
Im Greedy-Modus sendet jeder Knoten regelmäßig Nachrichten – so genannte Funkfeuer – an seine Nachbarn, die seine aktuelle Position enthalten. Basierend auf dieser Information allein werden Nachrichten weitergeleitet, indem für die Weiterleitung eines Pakets derjenige Knoten aus der Menge der Nachbarknoten gewählt wird, der die Distanz zum Zielknoten minimiert. In Abbildung 2-1 reicht beispielsweise Knoten x eine Nachricht, die für D bestimmt ist, an y weiter, weil y innerhalb der Sendereichweite von x die Distanz zu D minimiert. Falls von einem Knoten für längere Zeit kein Funkfeuer mehr empfangen wird, wird davon ausgegangen, dass dieser nicht mehr für das Weiterleiten von Nachrichten zur Verfügung steht. Der Speicherbedarf pro Knoten hängt allein von der Knotendichte im Netzwerk ab und ist unabhängig von der Gesamtzahl der Knoten im Netzwerk. Um den Aufwand für Funkfeuer gering zu halten, können auch reguläre Nachrichten, die ein Knoten versendet bzw. weiterleitet, von den Nachbarn für Aktualisierungen der Position des sendenden Knotens verwendet werden.

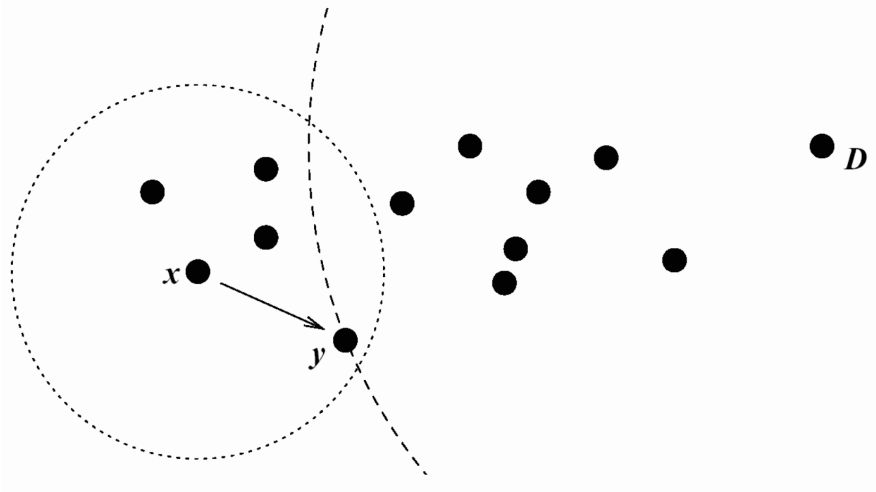

Abbildung 2-1. Der Greedy-Modus von GSPR. Für die Weiterleitung wird derjenige Knoten innerhalb der Sendereichweite (von x) ausgewählt, der die Distanz zum Ziel (D) minimiert. (Bild: [KK00])

Es gibt Fälle, in denen der Greedy-Modus bei der Weiterleitung versagt (Abb. 2-2). Der Knoten x möchte eine Nachricht an D weiterleiten, kann jedoch nicht direkt mit D kommunizieren. Obwohl es zwei mögliche Wege gäbe, um die Nachricht an D zu senden (x-y-z-D oder x-w-v-D), würde x im Greedy-Modus weder y noch w als Knoten für eine Weiterleitung der Nachricht akzeptieren, da sowohl y als auch w eine größere geografische Distanz zu D aufweisen als x selbst. Um dieses lokale Minimum zu umgehen, wechselt GPSR in den Perimeter-Modus. Im Perimeter-Modus werden Nachrichten nach der so genannten Rechten-Hand-Regel weitergeleitet. Die Rechte-Hand-Regel bewirkt, dass das Innere eines geschlossenen polygonförmigen Gebiets im Uhrzeigersinn durchlaufen wird. Wenn eine Nachricht von Knoten x bei Knoten y eintrifft, so wird sie auf derjenigen Verbindung weitergeschickt, die sich als erste - gegen den Uhrzeigersinn betrachtet - von der Verbindung x-y befindet. Diese Rechte-Hand-Regel kann dazu genutzt werden, um sich aus einem lokalen Minimum „herauszutasten“.

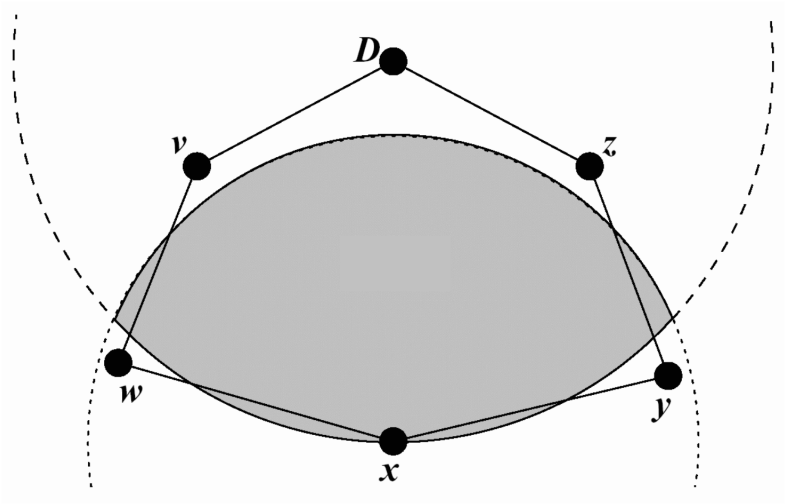

Abbildung 2-2. Obwohl es zwei Wege von x nach D gibt, versagt hier der Greedy-Modus. In diesen Fällen wird bei GPSR der Perimeter-Modus aktiviert. (Bild: [KK00])

Ein Problem bei Anwendung der Rechten-Hand-Regel ist, dass die Möglichkeit besteht, im Kreis zu laufen, wenn es Kanten im Graphen gibt, die sich kreuzen. Vor der Anwendung der Rechten-Hand-Regel erfolgt daher eine Planarisierung. Ein Graph, in dem sich keine zwei Kanten kreuzen, heißt planar. Zwei Beispiele für planare Graphen sind der *Relative Nachbarschaftsgraph* (RNG) und der *Gabriel-Graph* (GG) [Tous80, GS69]. Beim RNG wird eine Kante A-C genau dann entfernt, wenn sich noch (mindestens) ein Knoten B in der Schnittmenge der Sendekreise von A und C befindet. Beim GG wird eine Kante A-C entfernt, wenn sich noch (mindestens) ein Knoten B in dem Kreis befindet, dessen Mittelpunkt dem Mittelpunkt von A-C und dessen Durchmesser der Länge von A-C entspricht. Die Erzeugung dieser Graphen kann lokal durch die Knoten selbst erfolgen. Eine wichtige Eigenschaft dieser Graphen ist, dass – in der Theorie – die Entfernung von Kanten zur Erzeugung eines RNG oder GG das Netzwerk nicht partitioniert. In der Praxis kann es je nach Planarisierungsalgorithmus jedoch dazu kommen, dass das Netzwerk durchtrennt wird oder aber nicht alle sich kreuzenden Verbindungen entfernt werden. So entfernt im Prinzip ein Knoten C seine Verbindung zu einem weiter entfernten Knoten A, wenn C weiß, dass zwischen ihnen noch ein Knoten B liegt, der seine Nachrichten an A weiterleiten kann. Ob dies der Fall ist, berechnet C aus den Abständen der Knoten und den Senderadien. In realen Funknetzwerken können die Sendereichweiten jedoch nicht einfach durch perfekte Kreise substituiert werden und es gibt Hindernisse, die Kommunikationsverbindungen stören können. Ebenso können Lokalisierungsfehler oder die Mobilität der Knoten

Planarisierungsfehler verursachen. C kann zwar errechnen, dass B in der Lage sein müsste, Nachrichten an A weiterzuleiten, dies muss allerdings nicht tatsächlich der Fall sein, beispielsweise wegen eines Hindernisses, aufgrund nicht-idealer Sendereichweiten oder aufgrund eines Lokalisierungsfehlers. Wenn C aber dennoch die Kante A-C entfernt, wird das Netzwerk durchtrennt. Für dieses Problem wurden verschiedene Lösungsansätze vorgeschlagen, beispielsweise das *Cross-Link Detection Protocol* [KGKS05] oder das *Mutual Witness*-Protokoll [SHG04]. Beim Mutual Witness-Protokoll würde beispielsweise die Kante A-C nur entfernt werden, wenn A zuvor bestätigt hat, dass er tatsächlich mit B kommunizieren kann.

2.2 Stand der Technik

In diesem Abschnitt werden verschiedene bestehende Lösungen für die Dienst-Entdeckung vorgestellt. Hierbei werden einerseits Lösungen betrachtet, die von industriellen Anbietern entwickelt wurden und bereits auf dem Markt erhältlich sind, aber auf die hier betrachteten mobilen Roboternetzwerke nur eingeschränkt anwendbar sind. Sie haben jedoch die Terminologie und viele Grundelemente der aktuell noch in der Forschung befindlichen Lösungen geprägt. Im zweiten Teil wird dann auf Lösungen aus der Forschung eingegangen, die speziell für mobile Ad-hoc-Netzwerke entworfen wurden.

2.2.1 Klassische Protokolle zur Dienst-Entdeckung

In dynamischen, sowohl drahtgebundenen wie auch drahtlosen Netzwerken, in denen Teilnehmer regelmäßig ausfallen oder sich abmelden und neue Teilnehmer auftauchen können, haben Knoten im Allgemeinen kein Wissen über die im Netzwerk verfügbaren Ressourcen oder kein Wissen über den aktuellen Aufenthaltsort und die Erreichbarkeit einer Ressource. Das Entdecken von Ressourcen bzw. Diensten (aus Nutzersicht) ist eines der grundlegenden Probleme in derartigen Netzwerken. Eine besondere Herausforderung stellt dabei der zunehmende Übergang von immobilen, drahtgebundenen zu mobilen, drahtlosen Geräten in der Computer-, Unterhaltungs- und Kommunikationstechnik in den letzten Jahren dar. Von verschiedenen industriellen

Anbietern wurden erste Lösungen für das Entdecken von Ressourcen und Diensten in derartigen Netzwerken entwickelt, von denen drei in diesem Abschnitt vorgestellt werden.

Hier werden drei Protokolle, das *Bluetooth Service Discovery Protocol*, das *Jini Service Discovery Protocol* und die *UDDI*-Spezifikation (*Universal Description, Discovery, and Integration*) im Detail betrachtet, weil sie von besonderem Interesse für diese Arbeit sind. Das Bluetooth SDP ist von Bedeutung, weil die in dieser Arbeit für die Implementierung verwendeten Roboter auf Bluetooth basierende Kommunikationsmodule nutzen. Jini ist von Interesse, weil viele bisherige Projekte im Bereich der Roboternetzwerke Jini einsetzen. Und schließlich ist UDDI von Interesse, weil diese Spezifikation für die Dienst-Entdeckung von *Web Services* eingesetzt wird. In dieser Arbeit wird exemplarisch eine Web-Service-Schnittstelle in einen Roboter integriert (siehe 6.4). Zusätzlich repräsentieren die drei hier ausgewählten Protokolle auch drei Protokollklassen für verschiedene Netzwerkgrößen. Das Bluetooth SDP wurde für kleine drahtlose Netzwerke entworfen, Jini ist für mittelgroße gemischte Netzwerke geeignet und UDDI ist schließlich eine hochskalierbare Lösung für das Entdecken von Diensten im Internet. Man wird jedoch sehen, dass alle diese Lösungen Schwächen bei der Anwendung in drahtlosen, mobilen Multi-Hop-Netzwerken aufweisen.

2.2.1.1 Bluetooth Service Discovery Protocol

Bluetooth wurde als Industriestandard für die drahtlose Übertragung von Daten über kurze Distanzen entworfen [BSIG03]. Einer der Haupteinsatzgebiete von Bluetooth ist die Funktion als Kabelersatz für mobile Geräte und Zubehör wie Handys, PDAs, Drucker oder Fotokameras. Bis zu acht Bluetooth-Geräte können sich zu einem so genannten Piconetz zusammenschließen. In jedem Piconetz stellt ein Gerät den Master des Netzes dar, über den alle Kommunikationsverbindungen des Piconetzes laufen. Die anderen Teilnehmer des Piconetzes werden als *Slaves* bezeichnet. In der Bluetooth-Spezifikation wird angedacht, mehrere derartiger Piconetze zu einem so genannten Scatternetz zusammenzuschließen. Dabei fungieren einige Knoten der Piconetze als Brückenknoten und leiten Nachrichten von Netz zu Netz weiter. Da jeder Knoten zu einem bestimmten Zeitpunkt nur in einem Piconetz aktiv sein kann, müssen sich Brückenknoten in demjenigen Netz, in dem sie gerade nicht aktiv sind, in einen

Schlafmodus begeben. Die Bildung dieser Scatternetze und das Routen von Nachrichten in solchen Netzen werden von der Bluetooth-Spezifikation allerdings nicht abgedeckt.
Da Bluetooth für eine Vielzahl verschiedener Anwendungen und für mobile Umgebungen konzipiert wurde, haben die Entwickler ein einfaches Protokoll für die Dienst-Entdeckung integriert, das Bluetooth Service Discovery Protocol. Das Bluetooth SDP stellt einen einfachen Entdeckungsmechanismus zur Verfügung, bei dem ein Gerät, das einen Dienst sucht, alle anderen Geräte innerhalb seiner Sendereichweite nacheinander nach dem gewünschten Dienst befragt (Abb. 2-3). Das Bluetooth Service Discovery Protocol ist damit nicht auf größere Netzwerke skalierbar.

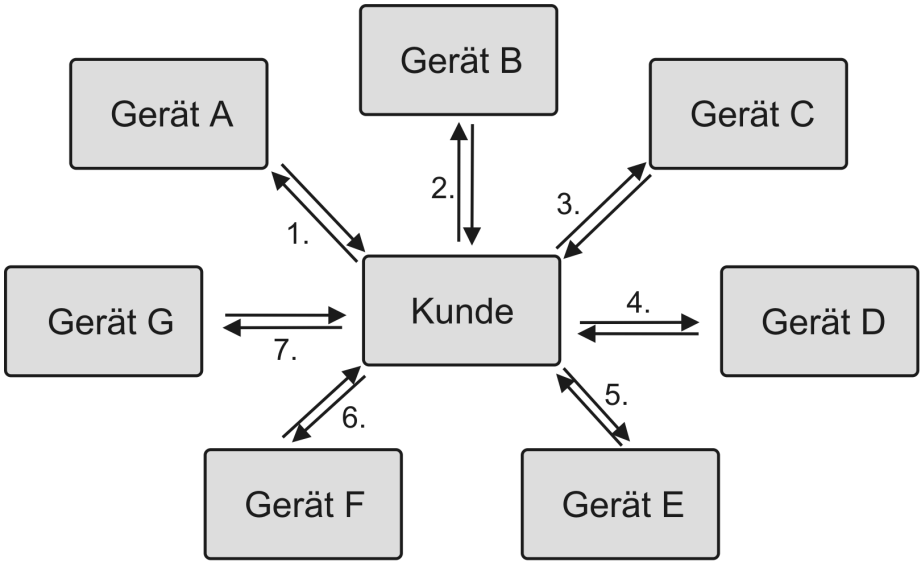

Abbildung 2-3. Beim Bluetooth Service Discovery Protocol befragen Kunden jedes Gerät innerhalb der Kommunikationsreichweite nacheinander nach den von ihm bereitgestellten Diensten.

2.2.1.2 Jini Service Discovery Protocol

Jini von *Sun Microsystems* ist eine Rahmenarchitektur für die Entwicklung von verteilten Anwendungen und basiert auf der Programmiersprache *Java* [Newm00]. Das Grundprinzip stellt dabei das Anbieten von Diensten im Netzwerk dar. Durch Standardisierung und Zugriff auf Dienste über dienstspezifische Schnittstellen, die vom Dienstleister selbst zur Verfügung gestellt werden, werden eine manuelle Konfiguration des Gesamtsystems und das Installieren von Treibern überflüssig.

Grundsätzlich kennt Jini drei verschiedene Typen von Teilnehmern: Dienstleister, Kunden, und so genannte *Lookup Services*. Lookup Services helfen Kunden und Dienstleister, sich im Netzwerk zu finden (Abb. 2-4).

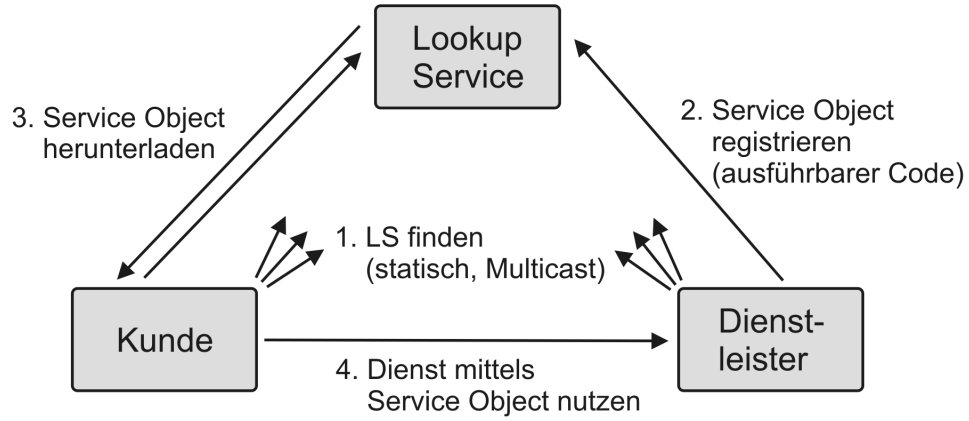

Abbildung 2-4. Beim Jini Service Discovery Protocol registrieren Anbieter ihre Dienste und die zugehörigen Zugriffsfunktionen bei wohlbekannten oder durch Multicast entdeckten *Lookup Services* im Netzwerk. Kunden können diese Lookup Services dann nach geeigneten Diensten befragen und die entsprechenden Zugriffsfunktionen herunterladen.

Wenn ein neuer Dienstleister im Netzwerk auftaucht, macht er sich bekannt, indem er für jeden Dienst, den er anbieten möchte, ein Java-basiertes *Service Object* an einen Lookup Service schickt. Ein Service Object ist der Teil eines Dienstes, der veröffentlicht wird und von den Kunden heruntergeladen und dazu genutzt wird, um auf diesen Dienst zuzugreifen. Hierzu muss der Dienstleister im ersten Schritt einen Lookup Service ausfindig machen. Dies kann auf zwei Arten erfolgen. Falls Identität und Aufenthaltsort eines Lookup Service bereits bekannt sind, beispielsweise weil in der betreffenden Arbeitsumgebung stets derselbe feste Lookup Service verwendet wird, kann dieser direkt kontaktiert werden. Wenn kein Lookup Service bekannt ist, wird eine Suchanfrage per Multicast ausgesendet und auf Antworten von Lookup Services gewartet. Wenn ein Lookup Service solch eine Suchanfrage erhält, sendet er ein Java-Objekt, einen so genannten *Registrar*, an den anfragenden Knoten zurück, den dieser dazu verwenden kann, um mit dem Lookup Service zu kommunizieren. Der Dienstleister verwendet dann diesen Registrar, um dem Lookup Service seine Dienste bekannt zu machen. Möchte nun ein Kunde einen Dienst nutzen, so versucht der Kunde ebenfalls zuerst einen Lookup Service zu kontaktieren. Über den Registrar, den der

Kunde vom Lookup Service zur Verfügung gestellt bekommt, kann der Kunde nach geeigneten Diensten fragen und die entsprechenden Service Objects herunterladen, um auf diese Dienste zuzugreifen. Das Service Object allein reicht aus, damit der Kunde den angeforderten Dienst nutzen kann. Ein Dienst kann dabei unterschiedliche Implementierungen von verschiedenen Herstellern haben.

Obwohl Jini für dynamische Netzwerkumgebungen ausgelegt wurde und damit implizit eine Eignung auch für mobile Knoten angedeutet wird, eignet es sich nur für Szenarien mit geringer Mobilität. Im Folgenden wird an zwei Punkten gezeigt, warum Jini nur bedingt für die hier betrachteten Roboternetzwerke geeignet ist. Bei hoher Mobilität der gewöhnlichen Knoten (die nicht als Lookup Service fungieren) müssten diese bei Verwendung von Jini jedes Mal, wenn sie sich in ein anderes Gebiet bewegen, per Multicast nach neuen Lookup Services suchen und sich dann neu bei diesen Lookup Service anmelden, was die Übermittlung eines kompletten Service Objects pro angebotenem Dienst beinhaltet. Alternativ können die mobilen Knoten auch versuchen, die Verbindung zu ihrem ursprünglichen Lookup Service aufrecht zu erhalten. Dieser kennt allerdings nicht die verfügbaren Dienste in der neuen Umgebung des Knotens, die für die Knoten in der Regel wichtiger sein werden als die verfügbaren Dienste in der Umgebung ihres Lookup Services. Außerdem kann die Kommunikationsentfernung der Knoten zu ihrem Lookup Service in diesem Fall beliebig groß werden.
Wenn Lookup Services mobil sind, sind die Auswirkungen noch gravierender. Grundsätzlich wird im Rahmen der Spezifikation nicht geklärt, wie neue Lookup Services entstehen. Stattdessen wird davon ausgegangen, dass Administratoren im Netzwerk Lookup Services in ausreichender Anzahl installiert haben. Bei Ausfall von Knoten, die einen Lookup Service anbieten, ist demnach nicht klar, wie diese Funktion automatisch von anderen Knoten übernommen werden kann. Wenn sich ein Lookup Service aus einem Gebiet fortbewegt, kennt es zwar die verfügbaren Dienste in seinem alten Gebiet, aber zuerst nicht die Dienste seines neuen Gebiets, die für die Knoten aus seiner neuen Umgebung aufgrund der räumlichen Nähe interessanter sind. Die Knoten seines alten Gebiets wiederum können versuchen, die Verbindung zu diesem Lookup Service aufrecht zu erhalten, was allerdings hohe Kommunikationskosten verursachen kann, da sowohl bei Anfragen als auch bei Bekanntmachungen von neuen Diensten der Lookup Service über eine potentiell weite Entfernung kontaktiert werden muss. Weiterhin kann dadurch die Situation entstehen, dass in einigen Gebieten eine sehr hohe

Dichte von Lookup Services auftritt, während in anderen Gebieten Lookup Services nicht in ausreichender Anzahl verfügbar sind. Von der Spezifikation ebenfalls nicht abgedeckt ist die Frage, ob Lookup Services untereinander Daten austauschen oder einander diese Aufgabe übergeben können.
Zusammenfassend lässt sich feststellen, dass Jini Mängel bei einem Einsatz in mobilen Multi-Hop-Netzwerken aufweist.

2.2.1.3 Universal Description, Discovery, and Integration (UDDI)

Die UDDI-Spezifikation (Universal Description, Discovery, and Integration) des OASIS Konsortiums beschreibt eine Lösung für das Entdecken von Diensten im Internet [OASI04]. Das Entdecken von Diensten mittels UDDI kann mit der Verwendung eines Telefonbuchs verglichen werden. Potentielle Kunden können sich an wohlbekannte UDDI-Anbieter wenden, um gewünschte Dienste zu lokalisieren und Interface-Beschreibungen dieser Dienste zu erhalten (Abb. 2-5). Mehrere derartige Anbieter können einen weltweiten Verbund bilden und ihre Daten auf regelmäßiger Basis austauschen. Sie können dabei sowohl als Verzeichnis wie auch als Index organisiert sein. Ein Verzeichnis ist dabei ein zentral kontrollierter Informationsspeicher. Um Beschreibungen von Diensten zu veröffentlichen, muss der Dienstanbieter diese Information explizit im Verzeichnis registrieren und der Besitzer des Verzeichnisses entscheidet, ob dieser Eintrag akzeptiert wird oder nicht. Beim Index-basierten Ansatz erfolgt die Bekanntmachung aus Sicht des Dienstanbieters passiv. Der Dienstanbieter stellt seine Dienstbeschreibungen im Netzwerk zur Verfügung und diejenigen Indexe, die sich dafür interessieren, sammeln sie ein. Das *World Wide Web Consortium* (W3C) empfiehlt Verzeichnisse für statische und kontrollierte Umgebungen, in denen sich die Informationen nicht häufig ändern [W3C04]. Indexe werden für Netzwerke empfohlen, die gut skalierbar sein sollen und bei denen eine große Vielfalt an Indizierungsstrategien notwendig sind. Bei der Verwendung von Indexen kann allerdings eine hohe Verzögerung auftreten zwischen dem Zeitpunkt, zu dem sich die Informationen zu einem Dienst ändern, und dem Zeitpunkt, zu dem sich diese Änderungen im Index widerspiegeln.
Die UDDI-Spezifikation wurde für die Dienst-Entdeckung im Internet entwickelt und eignet sich nicht für die Dienst-Entdeckung in mobilen Ad-hoc-Netzwerken. UDDI basiert auf wohlbekannten Servern und arbeitet stark zentralisiert. Die Spezifikation

weist die gleichen offenen Fragenstellungen auf wie bei Jini (2.2.1.2). So wird beispielsweise nicht geklärt, wie neue UDDI-Server entstehen oder wie bei Mobilität der Knoten eine gleichmäßige Verteilung der UDDI-Server im Operationsgebiet sichergestellt wird.

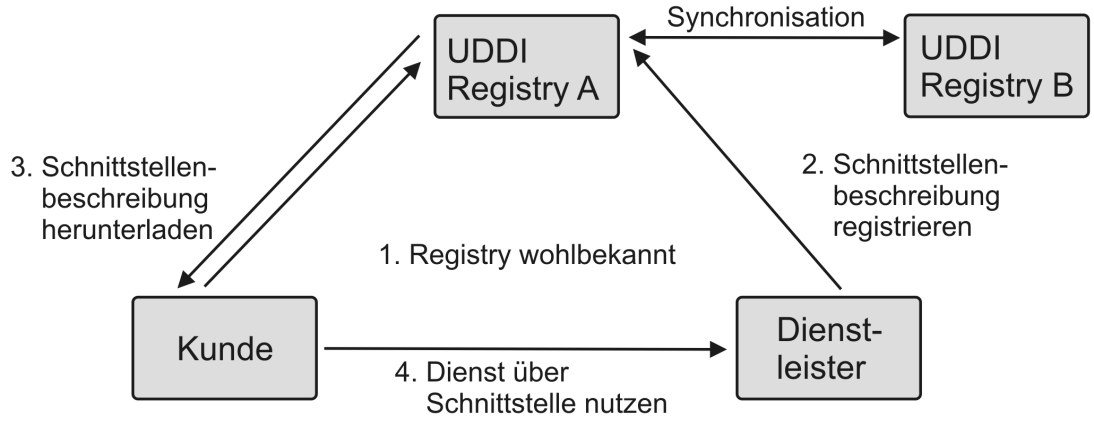

Abbildung 2-5. Bei Dienst-Entdeckungen basierend auf der UDDI-Spezifikation registrieren Dienstleister Schnittstellenbeschreibungen ihrer Dienste bei bekannten UDDI-Verzeichnissen (*Registry*). Kunden können diese Schnittstellenbeschreibungen herunterladen, um Dienste zu nutzen. Mehrere Verzeichnisse können einen Verbund bilden und ihre Daten auf regelmäßiger Basis austauschen.

2.2.2 Positionsbasierte Dienst-Entdeckung in mobilen Ad-hoc-Netzwerken

In diesem Überblick werden positionsbasierte Protokolle für die Dienst-Entdeckung in mobilen Ad-hoc-Netzwerken vorgestellt. Roboter navigieren in der realen Welt, womit sie ohnehin Positionsinformationen benötigen. Ebenso werden Roboter Dienste zum Teil physikalisch erbringen, beispielsweise bei Transportvorgängen. Dies setzt die Kenntnis der relativen Position der involvierten Roboter voraus und erfordert damit ein gemeinsames Koordinatensystem. Die Notwendigkeit von Positionsdaten ist somit keine wesentliche Einschränkung. Die Positionsdaten der Knoten können genutzt werden, um Eigenschaften der Protokolle wie Effizienz und Skalierbarkeit zu verbessern. Als einziges Verfahren, dass ohne Positionsdaten auskommt, wird das einfache Fluten des Netzwerks betrachtet. Dies stellt das einfachste Suchverfahren ohne großen Implementierungsaufwand dar und kann als Vergleichsbasis dienen. Eine

Übersicht anderer Protokolle für die Dienst-Entdeckung, die ohne Positionsinformation auskommen, findet sich in [Helm05].

2.2.2.1 Fluten

Fluten gehört zu den simpelsten Suchverfahren. Beim Fluten einer Suchanfrage wird jeder Knoten im Netzwerk nach dem gesuchten Dienst befragt. Beim einfachen Fluten verschickt jeder Knoten jede Nachricht genau einmal. Beim Empfang einer Nachricht überprüft jeder Knoten, ob die Nachricht neu ist oder ob er diese Nachricht schon einmal erhalten hat. Falls die Nachricht neu ist, sendet er sie genau einmal wieder aus. Bei n Knoten im Netzwerk werden also n Nachrichten verschickt (Abb. 2-6, Tabelle 2-1). Dies kann in funkbasierten Netzwerken zum *Broadcast Storm* [TNCS02] führen, bei dem innerhalb kürzester Zeit viele redundante Pakete verschickt werden, was in Kollisionen resultiert. Die Zeitdauer für Dienstsuchen ist beim einfachen Fluten proportional zu $\sqrt{n}$. Dies gilt unter den Annahmen, dass die Knoten sich über ein zweidimensionales Gebiet verteilen und dass sie beliebig mit ihren erreichbaren Nachbarn kommunizieren dürfen, also keine Kommunikationsstrukturen vorgegeben werden, die eingehalten werden müssen. In diesem Fall können Suchanfragen parallel im Netzwerk verbreitet werden und die maximale Zeitdauer für das Durchsuchen des Netzwerks ist proportional zur Entfernung von der Quelle der Suchanfrage zum aus ihrer Sicht weitesten Knoten im Netzwerk. Diese Distanz ist proportional zu $\sqrt{n}$, wenn sich die Knoten über ein zweidimensionales Gebiet verteilen.

Verschiedene Optimierungen sind denkbar, unter anderem eine expandierende Suche, bei der der Suchradius schrittweise erhöht wird, bis der gesuchte Dienst entdeckt oder eine maximale Anzahl von Suchschritten erreicht wird. Eine expandierende Suche bietet Vorteile, wenn der gesuchte Dienst im Mittel nah beim suchenden Knoten ist, ist jedoch ineffizienter, wenn die Knotenpaare weit entfernt im Suchgebiet verteilt sind.
Eine weitere Möglichkeit besteht darin, das Weiterschicken der Nachrichten effizienter zu gestalten. Dabei wiederholt nicht jeder Knoten die Nachricht, sondern nur ausgewählte Knoten. Dies verringert die Anzahl der versendeten Nachrichten, erhöht aber das Risiko, dass Knoten im Netzwerk die Nachricht nicht erhalten. Vorteile bietet dieses Verfahren vor allem, wenn die Knotendichte sehr hoch ist, und es somit bei

Anwendung des einfachen Flutens viele redundante Nachrichten gäbe. In [TNCS02] findet man eine Übersicht der gängigsten Verfahren, unter anderem basierend auf wahrscheinlichkeitsabhängigen Wiederholungen, dem Zählen von empfangenen Nachrichten, Knotenentfernungen oder Knotenpositionen. Beim probabilistischen Ansatz wiederholt ein Knoten eine Nachricht nur mit der Wahrscheinlichkeit p. Für p = 0 wird die Nachricht damit gar nicht weiterverschickt und für p=1 erhält man das einfache Fluten. Beim zählerbasierten Verfahren schickt ein Knoten eine empfangene Nachricht nicht sofort weiter, sondern wartet eine zufällige Zeitdauer ab und zählt während dieser Zeit mit, wie viele seiner Nachbarn diese Nachricht bereits wiederholt haben. Ab einer bestimmten Anzahl schickt er die Nachricht nicht mehr weiter, weil die erwartete zusätzliche Netzabdeckung nur gering ist. Beim distanzbasierten Verfahren schickt ein Knoten eine Nachricht nicht weiter, wenn er die Nachricht von einem geografisch nahen Knoten erhalten hat, weil die erwartete zusätzliche geografische Abdeckung nur gering ist. Bei ortsbasierten Verfahren wird schließlich die Positionsinformation der Knoten genutzt, um die zusätzliche Abdeckung zu berechnen. Wenn diese einen bestimmten Schwellwert unterschreitet, wird die Nachricht nicht wieder ausgesendet.

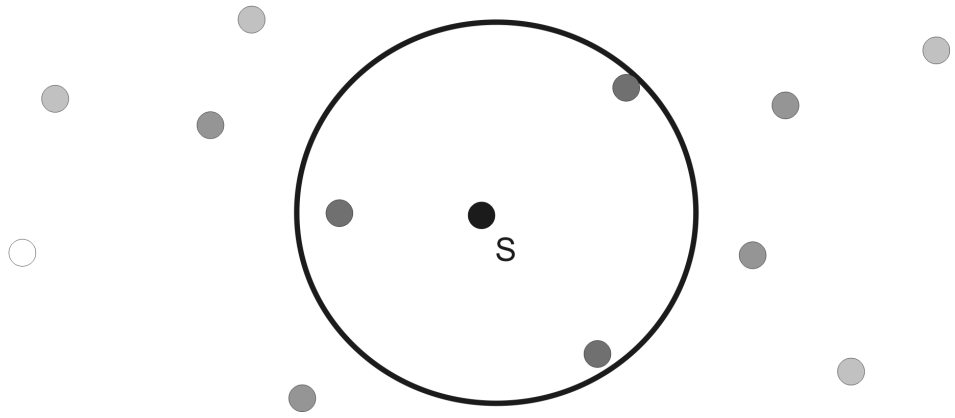

Abbildung 2-6. Beim Fluten sendet jeder Knoten eine empfangene, neue Nachricht genau einmal wieder aus. Hier wird illustriert, wie sich eine Nachricht von Knoten S im Netzwerk ausbreitet. Die unterschiedlichen Grauwerte zeigen an, wann ein Knoten die Nachricht frühestens empfängt (in Bezug auf die Anzahl der Weiterleitungen).

Die Vorteile des Flutens von Suchanfragen sind die Einfachheit und Robustheit des Verfahrens. Die Implementierung ist simpel, erfordert kaum Ressourcen in den Knoten

und es wird weitgehend sichergestellt, dass eine Suchanfrage alle Knoten im Netzwerk erreicht. Der wesentliche Nachteil ist das hohe Nachrichtenaufkommen in größeren Netzwerken und damit die mangelnde Skalierbarkeit des Verfahrens.

Tabelle 2-1. Nachrichtenaufwand und Latenz für das einfache Fluten.

	Installation / Dienst	Gesamtinstallation m Dienste/Knoten	Aktualisierung / Dienst	Suche / Dienst	Latenz / Suche	Speicher / Knoten
Fluten	-	-	-	n	$O(\sqrt{n})$	-

2.2.2.2 Grid Location Service (GLS)

Bei Anwendung von positionsbasiertem Routing muss die aktuelle Position des Zielknotens bekannt sein, bevor eine Kommunikationsverbindung zwischen zwei Knoten aufgebaut werden kann. Für diesen Zweck werden so genannte *Location Services* verwendet. In der Literatur wurden verschiedene Ansätze vorgeschlagen, die im Prinzip auch auf das Problem der Dienst-Entdeckung anwendbar wären [Camp05]. Einer der etablierten Ansätze, der *Grid Location Service* [LJCK+00], wird im Folgenden exemplarisch vorgestellt.

In diesem Absatz wird kurz das Grundprinzip von GLS erläutert, um dann im nächsten Absatz das Verfahren im Detail zu betrachten. GLS setzt voraus, dass jeder Knoten über eine global eindeutige Identifikationsnummer (ID) verfügt. Die Knoten werden geografischen Zellen zugeordnet, die hierarchisch so geordnet sind, dass jeweils vier Zellen eine Zelle der nächsthöheren Ordnung bilden. In jeder Ordnung rekrutiert ein Knoten B denjenigen Knoten S als Positionsserver für seine Positionsdaten, dessen ID numerisch am nächsten zu seiner eigenen ID B_{id} ist. Wenn nun ein anderer Netzwerkteilnehmer A den Knoten B finden möchte, sendet dieser eine Anfrage an denjenigen Knoten C unter den ihm bekannten Knoten, dessen ID C_{id} am nächsten zur derjenigen von B ist. Auf diese Weise wird die Suchanfrage im Netzwerk weitergeleitet – sie bewegt sich entlang einer Reihe von IDs, die B_{id} immer näher kommen – bis ein Server gefunden wird, der die aktuelle Position von B kennt. Es kann gezeigt werden, dass solch eine Suchanfrage immer zum Erfolg führt, wenn es keine Mobilität im

Netzwerk gibt, keine Knoten ausfallen und die Knoten genug Zeit hatten ihre Positionsserver zu rekrutieren.

GLS ist ein vollständig dezentralisiertes Verfahren. Jeder Knoten verfügt über eine kleine Menge von so genannten Location Servern, die seine aktuelle Position kennen und auf Anfrage anderen Knoten zur Verfügung stellen. Die Location Server eines Knotens sind über das Netzwerk verteilt und der Knoten aktualisiert sie regelmäßig mit seiner Position. Jeder Knoten dient selbst wiederum als Location Server für eine kleine Menge von Knoten des Netzwerks. Dabei ist die Dichte an Location Servern in der Nähe des jeweiligen Knotens höher als weiter entfernt vom Knoten.
Das Operationsgebiet wird geographisch in Form eines so genannten Quadtrees eingeteilt. Das Grundquadrat wird als Quadrat erster Ordnung bezeichnet. Vier Quadrate erster Ordnung bilden ein Quadrat der zweiten Ordnung, und so weiter. Jedes Quadrat n-ter Ordnung ist jedoch nur Teil genau eines Quadrates (n+1)-ter Ordnung, um Überlappungen zu vermeiden (Abb. 2-7). Das heißt aber auch, dass jeder Knoten genau einem Quadrat jeder Größe zugehörig ist. Die Autoren des Artikels weisen darauf hin, dass die Verwendung von Quadraten willkürlich ist und auch andere regelmäßige Formen denkbar wären. Diese Aufteilung des Operationsgebiets ist allen Knoten bekannt.

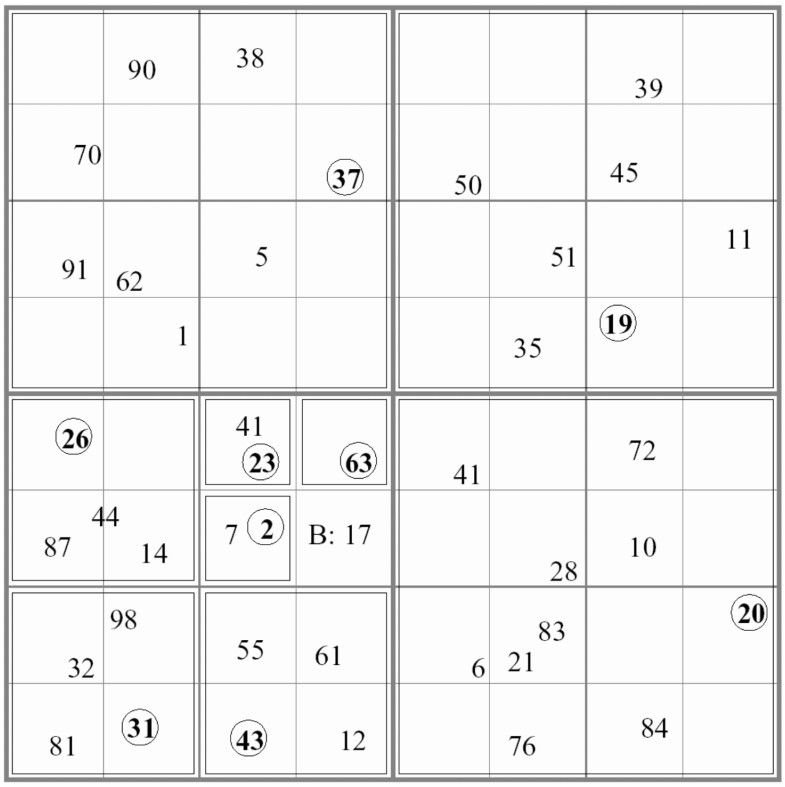

Abbildung 2-7. Beim Grid Location Service ist das Operationsgebiet in Form eines Quadtrees eingeteilt. Jeder Knoten wählt auf jeder Hierarchieebene drei Server, die seine aktuelle Position speichern. Als Beispiel sind hier die Server von Knoten B mit ID 17 hervorgehoben. (Bild: [LJCK+00])

Falls ein Knoten A einen Knoten B über einen positionsbasierten Routing-Algorithmus kontaktieren möchte, muss A einen Location Server von B finden und nach der aktuellen Position von B befragen. Dies bedeutet aber, dass B bei der Auswahl seiner Location Server die gleichen Knoten ausgewählt haben muss, die auch A bei seiner Suche verwenden wird. Die einzigen gemeinsamen Informationen über die A und B verfügen sind Bs ID und die geographische Aufteilung des Operationsgebiets in Form eines Quadtrees. Die numerischen Knoten-IDs werden dabei mittels einer Hashfunktion beispielsweise aus den Knotennamen berechnet. Die Hashfunktion muss sicherstellen, dass die IDs der Knoten gleichmäßig im ID-Raum verteilt sind.

Betrachten wir, wie Knoten B seine Location Server auswählt, die er mit seiner Position aktualisiert. Die Auswahl basiert ausschließlich auf seiner eigenen ID und der vorgegebenen Quadtree-Struktur, also nur den Informationen, über die auch die Knoten verfügen werden, die B später kontaktieren wollen. B rekrutiert diejenigen Knoten als Server für seine Positionsdaten, die IDs haben, die nahe der ID von B sind. Der ID-

Raum wird dabei als Ring betrachtet und die nächste ID bezüglich Bs ID wird definiert als die kleinste ID, die gerade noch größer als Bs ID ist. B wählt auf jeder Hierarchieebene drei Location Server für seine Positionsdaten in den benachbarten Quadraten. In jedem dieser Quadrate wählt B denjenigen Knoten als Server, der seiner ID am nächsten ist.
Möchte nun ein Knoten A mit B kommunizieren, muss er dazu vorher die aktuelle Position von B herausfinden. Wie findet Knoten A nun einen Location Server für Knoten B? A sendet eine Anfrage an denjenigen ihm bekannten Knoten C, der die nächste ID zu B_{id} hat. C sendet die Anfrage entsprechend weiter. Schließlich wird die Anfrage bei einem Location Server von B ankommen, der diese Anfrage an B weiterleiten kann. Wenn in dieser Anfrage die Position von A enthalten ist, kann B dann direkt eine Antwort zurückschicken. Es kann gezeigt werden, dass eine Anfrage immer zum Erfolg führt, wenn es keine Mobilität im Netzwerk gibt, wenn keine Knoten ausfallen und alle Knoten genug Zeit hatten, ihre Location Server zu rekrutieren [LJCK+00].

Bisher wurde eine Frage vernachlässigt: wie rekrutiert B seine Location Server, ohne alle Knoten im gesamten Operationsgebiet zu kennen und dann die geeigneten Location Server herauszupicken? Tatsächlich braucht B die Identität seiner Location Server gar nicht zu kennen. Die Bekanntmachung erfolgt ähnlich einer Suchanfrage. B sendet eine Bekanntmachung in denjenigen Quadraten, der aktualisiert werden soll. Innerhalb des Quadranten wird die Bekanntmachung wie bei einer Suchanfrage von Knoten zu Knoten weitergeleitet, die der ID von B immer näher kommen. Wenn in dem Quadranten alle Location Server bereits rekrutiert wurden, wird die Bekanntmachung schließlich stets beim geeigneten Knoten landet, nämlich dem Knoten, der die nächste ID bezüglich B innerhalb des Quadranten besitzt [LJCK+00]. Dieser wird dann als Location Server für B rekrutiert. Beim Initialisieren eines Netzwerks würden die Knoten zuerst ihre Location Server der zweiten Ordnung rekrutieren, dann könnten die Location Server dritter Ordnung rekrutiert werden, usw. Location Server erster Ordnung werden nicht benötigt. Für die Grundkommunikation mit direkt benachbarten Konten wird bei GLS ein so genanntes *Two Hop Distance Vector Protocol* verwendet. Dabei kennen alle Knoten ihre Nachbarn und die Nachbarn ihrer Nachbarn, also die Topologie des Netzwerks bis zu einem Abstand von zwei Sprüngen. Dies stellt bei GLS sicher,

dass ein Knoten alle Knoten seines Grundquadranten (bei den von den Autoren des Artikels gewählten Zellgrößen und Senderadien) kennt [LJCK+00].

Ein Knoten sendet eine Aktualisierung aus, wenn er sich eine bestimmte Distanz bewegt hat. Dabei sind die Knoten, die er benachrichtigt von der zurückgelegten Strecke abhängig. Die Knoten in seiner eigenen Zelle braucht er nicht zu benachrichtigen, da sie in seiner Zwei-Hop-Nachbarschaft liegen. Allgemein werden die Knoten der Quadrate n-ter Ordnung in das Update eingeschlossen, wenn der Knoten sich um d * 2^(n-2) weiterbewegt hat. Bewegt sich der Knoten also beispielsweise um die Distanz d weiter, so informiert er alle Knoten in den von ihm aus betrachteten Quadraten zweiter Ordnung (d * 2^(n-2) = d für n = 2). Bewegt er sich um 2d weiter, informiert er auch die Knoten in den Quadraten dritter Ordnung (d * 2^(n-2)*d = 2d für n = 3). Um auch Knoten, die sich nicht bewegen, im Netzwerk bekannt zu machen, senden Knoten unabhängig von der bewegten Strecke auch zeitbasiert Aktualisierungen mit einer niedrigen Frequenz aus.

Wenn ein Knoten eine Zelle verlässt, dauert es eine Zeit lang, bis seine neue Position im gesamten Netzwerk bekannt ist. Für diesen Fall schlagen die Autoren so genannte *Forwarding Pointer* vor, die ein Knoten hinterlässt, bevor er eine Zelle verlässt. Ein Forwarding Pointer enthält die neue Aufenthaltszelle des Knotens. Nachrichten, die noch in der alten Zelle eintreffen, werden entsprechend weitergeleitet. Forwarding Pointer sind zellengebunden, d.h. neue Knoten einer Zelle erhalten die jeweils aktuellen Forwarding Pointer der Zelle, während Knoten, die eine Zelle verlassen, die Forwarding Pointer ihrer alten Zelle vergessen.
Neben ihrer Positionstabelle, die die Position derjenigen Knoten enthält, für die ein Knoten als Location Server zuständig ist, gibt es noch den Positionscache, der Informationen enthält, die der Knoten durch Auswerten weitergeleiteter Bekanntmachungen und Aktualisierungen lernt. Diese Informationen werden mit einer kurzen Ablaufzeit versehen. Der Cache wird nur für Suchen genutzt, die vom Knoten selbst ausgehen, nicht für Suchen, die von dem Knoten weitergeleitet werden, da dies sonst das eigentlich vorgesehene Suchverfahren von GLS unterlaufen würde.

Tabelle 2-2 stellt die Kommunikations-, Speicher- und Zeitkomplexität von GLS dar. Es werden die gleichen Annahmen wie in 4.2 getroffen. Installation pro Dienst gibt die

Anzahl der Nachrichten an, die ein neu eingetroffener Dienst benötigt, um sich im Netzwerk bekannt zu machen. Bei GLS wird an jede der log(n) Hierarchieebenen eine Bekanntmachung durch das Netzwerk geschickt. Hieraus ergeben sich die in der Tabelle angegebenen Kosten. Gesamtinstallation gibt die Gesamtzahl an Nachrichten an, die benötigt werden, um ein Netzwerk aus n Knoten zu initialisieren. Wenn jeder Knoten im Mittel m Dienste anbietet, müssen n*m Dienste bekannt gemacht werden. Dies multipliziert mit dem Aufwand für die Bekanntmachung eines Dienstes ergibt die Kosten für die Gesamtinstallation. Aktualisierung pro Dienst gibt die Anzahl der Nachrichten an, die benötigt werden, um Aktualisierungen eines Dienstes im Netzwerk zu verbreiten, z. B. bei Positionsänderungen. Die Kosten hierfür sind geringer als bei der Installation, weil GLS bei einer Aktualisierung selektiv vorgeht und nur ausgewählte Positionsserver in Abhängigkeit von der zurückgelegten Strecke des jeweiligen Knotens informiert. Suche pro Dienst gibt die Anzahl der Nachrichten an, die benötigt werden, um einen Dienst im Netzwerk zu lokalisieren. Bei GLS durchläuft eine Suche bis zu O(log(n)) im Netzwerk verteilte Positionsserver, bis der gewünschte Knoten lokalisiert ist. Die Latenz gibt die Zeitdauer an, die benötigt wird, um einen Dienst zu lokalisieren. Sie ist proportional zu den Kosten für eine Dienstsuche, weil GLS nur eine einzelne Suchnachricht versendet und die Latenz proportional zu der Anzahl der benötigten Nachrichtensprünge ist. Schließlich gibt Speicher pro Knoten an, wie viele Speicherplätze für Dienste ein Knoten des Netzwerks bereitstellen muss. Bei GLS rekrutiert jeder Knoten O(log(n)) Positionsserver. Mit m Diensten pro Knoten ergibt sich der in der Tabelle angegebene Speicherbedarf pro Knoten.

Tabelle 2-2. Nachrichten- und Speicherkomplexität sowie Latenz von GLS.

	Installation / Dienst	**Gesamtinstallation m Dienste/Knoten**	**Aktualisierung / Dienst**	**Suche / Dienst**	**Latenz / Suche**	**Speicher / Knoten**
GLS	$O(\sqrt{n} \log n)$	$O(nm\sqrt{n} \log n)$	$O(\sqrt{n})$	$O(\sqrt{n} \log n)$	$O(\sqrt{n} \log n)$	$O(m \log n)$

Ein Vorteil von GLS ist seine vollständige Dezentralisierung. Damit werden alle Knoten bei einer angenommenen Gleichverteilung der Knoten und Anfragen im Mittel gleich stark belastet und es gibt keine Flaschenhälse im Netzwerk. Auch das

Lokalitätskriterium wird teilweise eingehalten: Die höhere Dichte an Location Servern in der Nähe des Knotens stellt sicher, dass nahe Anfragen auch tatsächlich von nahen Location Servern beantwortet werden. Einerseits werden dadurch Suchanfragen effizient durchgeführt, andererseits gewährleistet dies auch, dass im Fall von Netzwerk-Partitionierungen die einzelnen Partitionen weiterhin funktionsfähig bleiben.

Wenn die Existenz einer Hashfunktion angenommen wird, die Dienstbeschreibungen auf global eindeutige IDs abbildet, unterscheidet sich auf den ersten Blick die Suche nach einem Knoten nicht wesentlich von der Suche nach einem Dienst. Im Folgenden wird anhand von GLS gezeigt, wie Unterschiede zwischen der Lokalisierung von Knoten und der Lokalisierung von Diensten eine direkte Anwendung von Hash-basierten Location Services auf das Problem der Dienst-Entdeckung einschränken. Dabei sei vorangestellt, dass GLS von seinen Autoren explizit als Location Service, also zur Positionsbestimmung von Knoten repräsentiert in Form von eindeutigen, numerischen IDs entwickelt wurde – und nicht für die Dienst-Entdeckung, die unterschiedliche Anforderungen und Randbedingungen hat. Die im Folgenden genannten Schwächen stellen also keine Entwurfsfehler dar.
Einerseits kann in Frage gestellt werden, ob es sinnvoll und möglich ist, Dienstbeschreibungen auf IDs abzubilden, da damit erhebliche Nachteile bezüglich der Flexibilität und Komplexität der Anfragen verbunden sind [CR03, KKO03]. Eine Anfrage folgender Art wäre damit beispielsweise nicht möglich: “Wir suchen einen Roboter der 75 Kilogramm bei einer Minimalgeschwindigkeit von 10 km/h transportieren kann. Dabei ist die Geschwindigkeitsanforderung wichtiger und wir könnten eine verminderte Tragfähigkeit bis auf 50 kg akzeptieren, um die gewünschte Geschwindigkeit zu erreichen.“ Weiter gibt es zwei noch wesentlichere Unterschiede zwischen der Lokalisierung von Diensten und der Lokalisierung von Knoten in einem Netzwerk. Gerade in größeren Netzwerken wird ein Dienst nicht einzigartig sein und Dienstleister werden nicht nur einen einzigen, sondern verschiedene Dienste anbieten. Dienstleister werden also sich überschneidende Mengen von Diensten anbieten. Wenn GLS unter diesen beiden Randbedingungen betrachtet wird, stellen sich zwei wesentliche Nachteile heraus. Erstens würde sich das Nachrichtenaufkommen bezüglich der Nachrichtenanzahl wesentlich erhöhen, da jeder einzelne Dienst eines Knotens seine eigene ID hätte und damit auf seine eigenen spezifischen Server abgebildet werden müsste. Zweitens würden alle identischen Dienste mit der gleichen ID zu den gleichen

Servern im Netzwerk abgebildet. Mehrere Instanzen eines Dienstes würden nicht zur Robustheit des Systems beitragen, wenn die entsprechenden Server ausfallen sollten. Damit hat GLS nur eine eingeschränkte Anwendbarkeit auf das Problem der Dienst-Entdeckung.

2.2.2.3 Lightweight Overlay for Service Discovery in MANETs

Das Grundprinzip von *LANES (Lightweight Overlay for Service Discovery in Mobile Ad hoc Networks)* [KKO03] basiert auf einer Struktur aus vertikalen Bahnen. Entlang dieser Bahnen werden Bekanntmachungen von Diensten verschickt, Suchanfragen wiederum werden horizontal von Bahn zu Bahn weitergeleitet (Abb. 2-8). Jeder Knoten des Netzwerks ist Mitglied genau einer Bahn. Innerhalb jeder Bahn sind die Knoten eindeutig in Nachfolger und Vorgänger geordnet. Alle Knoten einer Bahn kennen sich untereinander und die von ihnen angebotenen Dienste. Von außen betrachtet können die einzelnen Knoten einer Bahn demnach als äquivalent betrachtet werden. Wenn angenommen wird, dass die n Knoten eines Netzwerks gleich auf einer quadratischen Grundfläche verteilt sind, werden Dienstbekanntmachungen im Mittel an $\sqrt{n}$ Knoten geschickt. Wenn ein gesuchter Dienst nicht in der eigenen Bahn zur Verfügung steht, werden Suchanfragen an beliebige Knoten in den horizontal benachbarten Bahnen geschickt.

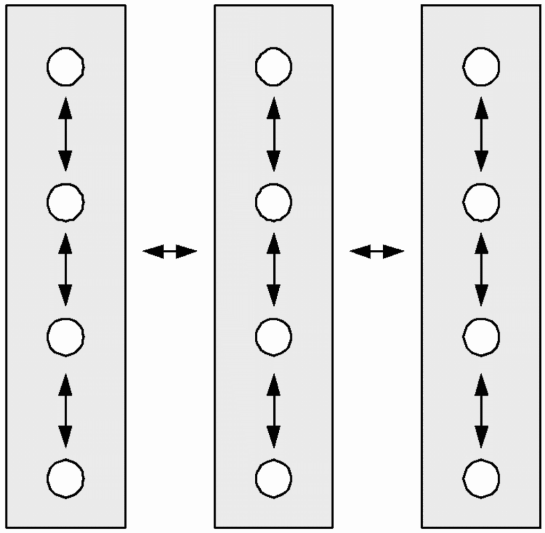

Abbildung 2-8. Bei LANES sind die Knoten in vertikalen Bahnen organisiert, in denen sich alle Knoten untereinander kennen. Suchen werden horizontal von Bahn zu Bahn weitergeleitet. (Bild: [KKO03])

Die Anmeldung eines Knoten bei einer Bahn hat folgenden Ablauf. Ein Knoten N, der einer Bahn beitreten möchte, sendet eine Teilnahmeanfrage an alle Knoten innerhalb seiner Sendereichweite. Jeder Knoten X, der solch eine Teilnahmeanfrage empfängt, schickt ein Teilnahmeangebot bestehend aus seiner eigene Adresse, der Adresse seines oberen Nachbarn (als X.T bezeichnet) und der aktuellen Länge seiner Bahn (soweit bekannt) als Antwort. N sammelt die Angebote und wählt eines aus. Dabei bevorzugt N Angebote aus kurzen Bahnen oder Angebote, bei denen N Teilnahmeangebote sowohl von X als auch X.T erhalten hat. N schickt eine Zustimmung zur Teilnahme an den gewählten Knoten X* und dessen oberen Nachbarn X*.T, worauf X* mit einer Bestätigung antwortet. Diese Bestätigung enthält die obere Hälfte des von X* verwalteten Gebiets und die von X* gespeicherten Dienstbeschreibungen aller Knoten der Bahn. N fügt sich damit in die Bahn zwischen X* und X*.T ein. X* und X*.T informieren nun auch ihre Vorgänger bzw. Nachfolger in der Bahn über den neu hinzugekommenen Knoten N.

Die Suche nach einem Dienst ist nun sehr einfach. Jeder Knoten kennt alle verfügbaren Dienste in seiner Bahn. Falls der gesuchte Dienst nicht in seiner eigenen Bahn verfügbar ist, befragt der Knoten seine benachbarten Bahnen. Die beiden Anfragen werden so lange von Bahn zu Bahn weitergeleitet, bis der gewünschte Dienst gefunden wird oder die Anfragen an die Grenzen des Netzwerks stoßen.

Durch die Dynamik des Netzwerks kann es zu unterbrochenen Verbindungen in Bahnen kommen. Um unterbrochene Verbindungen in einer Bahn zu detektieren, müssen diese zunächst detektiert werden können. Dies erfolgt durch regelmäßige Nachrichten, die jeder Knoten X an seinen oberen Nachbarn X.T sendet und die Adressen von X und X.B (dem unteren Nachbarn von X) enthält. Falls solche Nachrichten ausbleiben, geht ein Knoten X* davon aus, dass er die Verbindung zu X*.B verloren hat und versucht X*.B.B zu kontaktieren. Ist dieser Kontaktversuch erfolgreich, wird die Bahn durch eine direkte Verbindung zwischen X* und X*.B.B wieder hergestellt. Weiterhin werden alle vom nicht mehr kontaktierbaren Knoten X*.B angebotenen Dienste in der Bahn als nicht mehr verfügbar gekennzeichnet und entfernt. Falls auch X*.B.B nicht kontaktierbar ist, wird angenommen, dass das Netzwerk partitioniert ist und die Bahn wird aufgeteilt. In dem Fall informiert X* die restlichen kontaktierbaren Knoten seiner Bahn über die Partitionierung und die von den Knoten des nicht mehr kontaktierbaren Teils der Bahn angebotenen Dienste werden aus der Liste der verfügbaren Dienste entfernt. Die Sender dieser Nachrichten detektieren Fehlerfälle auf die gleiche Art. Wenn X* keine Nachricht an X*.T schicken kann, nimmt X* an, dass X*.T die Bahn unabsichtlich verlassen hat und wartet auf eine Nachricht von X*.T.T. Wenn diese Nachricht innerhalb einer gewissen Zeit eintrifft, wird die Bahn durch eine direkte Verbindung zwischen X* und X*.T.T repariert. Ansonsten geht X* von einer Partitionierung des Netzwerks aus und geht entsprechend dem oben beschriebenen Fall vor.

Wenn getrennte Netzwerkpartitionen in Kontakt miteinander treten, erfolgt eine Verschmelzung der Partitionen. Detektiert wird solch ein neuer Kontakt mittels zufällig generierter Partitionsnummern, die jeweils gleich für alle Knoten einer Partition sind und in ihnen verbreitet werden. Wenn ein Knoten von einem anderen Knoten der gleichen Bahn aber mit einer unterschiedlichen Partitionsnummer hört, wird eine Verschmelzung eingeleitet. Dabei tauschen die Bahnen Informationen über ihre Dienste miteinander aus und werden zu einer Bahn.

Tabelle 2-3 stellt die Kommunikations-, Speicher- und Zeitkomplexität von LANES dar. Es werden die gleichen Annahmen wie in 4.2 getroffen. LANES verschickt Bekanntmachungen in vertikaler Richtung bis an beide Enden des Netzwerks, die Kosten der Installation und der Aktualisierung pro Dienst betragen damit O($\sqrt{n}$). Bei m

Diensten pro Knoten, deren Beschreibungen im Fall von LANES jedoch zusammengefasst innerhalb der Bahn verbreitet werden können, und n Knoten im Netzwerk ist der Aufwand für die Gesamtinstallation proportional zu $n\sqrt{n}$ (Anzahl der Nachrichten multipliziert mit der mittleren Anzahl der Nachrichtensprünge). Für Dienstsuchen verschickt LANES horizontale Suchnachrichten durch das Netzwerk. Damit entsprechen die Kosten von Dienstsuchen denen der Installation. Dies gilt ebenso für die Latenz, da für eine Suche nur eine einzelne Nachricht versendet wird und die Latenz proportional zu der Anzahl der benötigten Nachrichtensprünge ist. Die Bekanntmachung eines Dienstes wird bei LANES an $O(\sqrt{n})$ Knoten geschickt. Wenn jeder Knoten in etwa m Dienste anbietet, muss jeder Knoten des Netzwerks $O(m\sqrt{n})$ Dienste speichern.

Tabelle 2-3. Nachrichten- und Speicherkomplexität sowie Latenz von LANES.

	Installation / Dienst	**Gesamtinstallation m Dienste/Knoten**	**Aktualisierung / Dienst**	**Suche / Dienst**	**Latenz / Suche**	**Speicher / Knoten**
LANES	$O(\sqrt{n})$	$O(n\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(m\sqrt{n})$

Ein Vorteil des LANES-Ansatzes ist seine völlige Dezentralisierung. Weiterhin kann LANES Knoten, die mehr als nur einen Dienst anbieten, effizient unterstützen, da die Bekanntmachungen der Dienste in einer Nachricht gebündelt werden können.

Ein Nachteil von LANES ist, dass gefundene Dienste beliebig weit entfernt sein können, auch wenn ein näherer Dienst verfügbar wäre. So wird eine Suchanfrage nicht mehr weitergeleitet, sobald ein geeigneter Dienst gefunden wurde. Allerdings kann dieser gefundene Dienst an jeder Position der Bahn sein, beispielsweise am oberen oder unteren Ende der Bahn. Ein äquivalenter Dienst, der sich auf der gleichen Höhe wie der Dienstnutzer nur eine Bahn weiter befände, würde nicht gefunden werden. Der wesentliche Nachteil des LANES-Ansatzes ist jedoch die kostspielige Wartung der Bahnenstruktur, wenn Knoten sich bewegen. Knoten, die in Bewegung sind, insbesondere Knoten, die sich quer durch das Netzwerk bewegen, können ein hohes Nachrichtenaufkommen auslösen. Auf der einen Seite wird sich der Knoten neuen Bahnen anschließen, deren Daten zwischenspeichern und die Bahnen wieder verlassen. Noch kostspieliger ist jedoch, dass das Lokalitätskriterium vernachlässigt wird, um die

Konsistenz in den vertikalen Bahnen aufrecht zu erhalten. Jede Bahn, die von dem Knoten besucht wird, wird Informationen über diesen Knoten entlang seiner gesamten Höhe verbreiten. So wird schon bei einem kleinen Anteil sich bewegender Knoten das gesamte Netzwerk nicht zur Ruhe kommen können. Bei beispielsweise n = 400 Knoten im Netzwerk, die – im Idealfall für LANES – etwa gleich auf einer ungefähr quadratischen Grundfläche im Netzwerk verteilt sind, würden etwa $\sqrt{n}$ = 20 Knoten eine Bahn bilden. Das heißt LANES würde versuchen, etwa 20 solcher Bahnen, die sich über etwa jeweils 20 Sprünge durch das Netzwerk erstrecken, konsistent zu halten.

2.2.2.4 Geography-based Content Location Protocol (GCLP)

Beim *Geography-based Content Location Protocol* [TV04] senden Knoten periodisch Bekanntmachungen aus, die die von ihnen angebotenen Dienste enthalten. Diese Bekanntmachungen werden horizontal und vertikal im Netzwerk verbreitet (Abb. 2-9). Knoten, die eine Bekanntmachung erhalten oder in ihrer Umgebung mithören, speichern die darin enthaltenen Dienste in ihrem Cache. Suchanfragen werden in GCLP ebenfalls horizontal und vertikal durch das Netzwerk geleitet. Es besteht eine hohe Wahrscheinlichkeit, dass sich die Trajektorien einer Suchanfrage und die einer passenden Bekanntmachung im Netzwerk in einem oder mehreren Knoten kreuzen. Wenn ein Knoten eine Suchanfrage erhält oder mithört und er die Anfrage aus seinem Cache beantworten kann, dann sendet er eine Antwort an den suchenden Knoten.

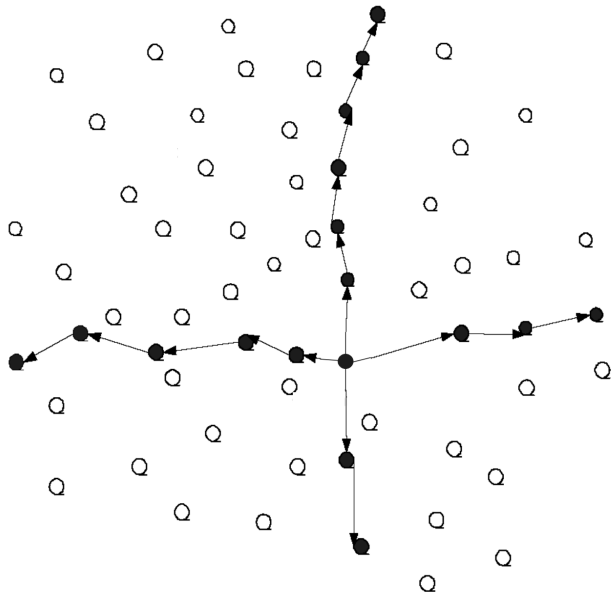

Abbildung 2-9. Bei GCLP werden Bekanntmachungen von Diensten und Suchanfragen horizontal und vertikal im Netzwerk verbreitet. (Bild basiert auf [TV04])

Wenn es mehrere gleiche Dienste im Netzwerk gibt, dann wird von einem Knoten nur diejenige Bekanntmachung weitergeleitet, die ihm geografisch am nächsten ist.
Damit Knoten die Bekanntmachungen horizontal und vertikal im Netzwerk weiterleiten können, müssen sie ihre Position und die ihrer Nachbarn kennen. Dies wird durch das regelmäßige Versenden von Hallo-Nachrichten an Nachbarknoten sichergestellt. Die Hallo-Nachrichten enthalten die Identität des Senders und dessen Position.

Tabelle 2-4. Nachrichten- und Speicherkomplexität sowie Latenz von GCLP.

	Installation / Dienst	**Gesamtinstallation m Dienste/Knoten**	**Aktualisierung / Dienst**	**Suche / Dienst**	**Latenz / Suche**	**Speicher / Knoten**
GCLP	$O(\sqrt{n})$	$O(n\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(m\sqrt{n})$

Die Analyse für LANES ist in dieser Form auch weitgehend für GCLP gültig. Tabelle 2-4 stellt die Kommunikations-, Speicher- und Zeitkomplexität von GCLP dar. Die Kosten von GCLP unterscheiden von LANES lediglich durch Konstanten und haben somit keine Auswirkungen auf die Tabelle. Vorteile des GCLP-Ansatzes sind - wie auch im Fall von LANES - seine völlige Dezentralisierung und die effiziente

Unterstützung von Knoten, die mehr als nur einen Dienst anbieten, da die Bekanntmachungen einfach gebündelt werden können. Zusätzlich wird im Unterschied zu LANES sichergestellt, dass bei mehreren gleichen Diensten im Netzwerk tatsächlich ein naher Dienst lokalisiert wird. Ein Nachteil von GCLP ist, dass Suchanfragen stets bis ans Ende des Netzwerks geleitet werden. Alle Knoten dürfen ja auf eine Suchanfrage antworten, wenn sie sie mithören. Das bedeutet, dass diejenigen Knoten, die die Suchanfrage weiterleiten, nicht mit Bestimmtheit wissen, ob die Anfrage nicht schon von einem anderen Knoten in ihrer Umgebung beantwortet wurde. Das Mithören von Suchantworten allein genügt nicht, denn es gibt Knotenkonstellationen, bei denen die für die Weiterleitung zuständigen Knoten selbst im statischen Fall, also ohne Mobilität der Knoten, Suchantworten bereits nicht mithören können. Und wie auch bei LANES werden Bekanntmachungen im schlechtesten Fall nach wie vor entlang der gesamten Höhe und Breite des Netzwerks propagiert. Insbesondere Knoten, die sich quer durch das Netzwerk bewegen, können so ein hohes Nachrichtenaufkommen auslösen. Dies wird dadurch verstärkt, dass die Organisationsstruktur auf einzelnen Knoten und nicht auf Gruppen von Knoten basiert und dass sowohl Bewegungen in horizontaler wie auch vertikaler Richtung diesen Effekt haben.

2.2.2.5 Rendezvous Regions (RR)

Rendezvous Regions ist ein etabliertes Beispiel für einen Lösungsansatz basierend auf geografischem Hashing [SH04a]. Das Netzwerk wird geografisch in rechteckige Zellen, so genannte Rendezvous Regions, aufgeteilt. Ressourcen im Netzwerk verwenden eine global bekannte Hashfunktion, um ihre Heimatregion zu bestimmen, in der sie Informationen über ihren aktuellen Aufenthaltsort speichern können. Innerhalb einer solchen Rendezvous Region werden einige Knoten als Server gewählt, die für die Verwaltung dieser Positionsdaten verantwortlich sind. Sie beantworten Anfragen anderer Knoten, die eine Ressource suchen, deren Positionsinformationen in diese Rendezvous Region abgebildet wurde (Abb. 2-10).

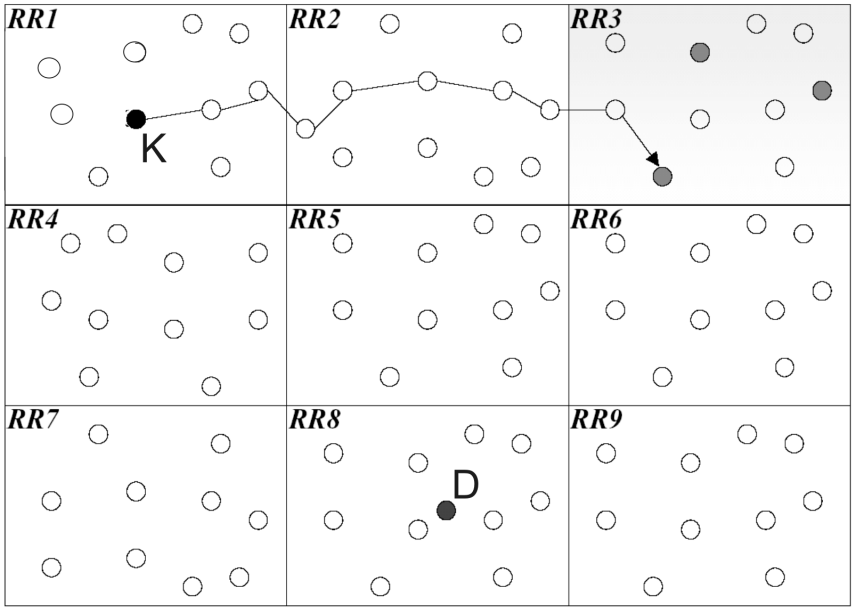

Abbildung 2-10. Rendezvous Regions basiert auf geografischem Hashing. Dienste im Netzwerk verwenden eine global bekannte Hashfunktion, um ihre Heimatregion zu bestimmen, in der sie Informationen über ihren aktuellen Aufenthaltsort speichern. Der Knoten K nutzt auf der Suche nach Dienst D dieselbe Hashfunktion, um das Heimatgebiet von D zu bestimmen (RR3) und dann dortige Knoten nach dem aktuellen Aufenthaltsort von D zu befragen. (Bild basiert auf [SH04a])

Die Voraussetzungen für Rendezvous Regions sind, dass das Operationsgebiet im Voraus bekannt ist und die Knoten die Möglichkeit haben, sich absolut im Operationsgebiet zu lokalisieren.

Die Server, die in einer Rendezvous Region Suchanfragen anderer Knoten beantworten, werden soweit erforderlich erst beim Eintreffen neuer Positionsdaten eines Dienstes gewählt. Wenn eine Bekanntmachung in der Zielregion eintrifft, flutet der erste Knoten, der die Bekanntmachung erhält (der so genannte *Flooder*), die enthaltenen Informationen in der Region. Jeder Server dieser Region, der diese Informationen empfängt, sendet eine Bestätigung an den Flooder. Wenn der Flooder nach einer bestimmten Zeit nicht genügend derartiger Bestätigungen empfängt, flutet er die Nachricht erneut, fügt diesmal jedoch eine Zahl $p \in [0,1]$ hinzu, die die Wahrscheinlichkeit ausdrückt, dass sich ein Knoten in der Region neu zum Server ernennt. Wenn wiederum nicht genügend Bestätigungen zurückkommen, wird p solange erhöht, bis p=1 ist oder eine genügend große (durch den Entwickler festgelegte) Anzahl von Bestätigungen beim Flooder eintreffen. Wenn Server eine Region verlassen oder ausfallen, werden auf diese Weise neue Server nachbestimmt. Die neuen Server

informieren sich bei den alten Servern über die alten gespeicherten Schlüssel. Der Flooder schickt nach erfolgreich durchgeführter Einfügeoperation schließlich eine Bestätigungsnachricht an den Knoten, der die Position seines Dienstes in diese Region abbilden wollte.

Das Finden eines Dienstes erfolgt ähnlich einer Bekanntmachung. Durch die global bekannte Hashfunktion wird der gesuchte Dienst auf eine Region abgebildet und eine Suchanfrage in diese Region gesendet. Der erste Knoten in der Region, der diese Suchanfrage erhält, schickt diese weiter an einen beliebigen Server, der ihm bekannt ist und leitet dessen Antwort zurück an den anfragenden Knoten. Falls der erste Knoten noch keinen Server in seiner Region kennt, flutet er die Anfrage in der Region.
Um in Zeiten niedrigen Nachrichtenaufkommens den Ausfall aller Server zu vermeiden, verfügt jeder Server über einen Timer, der bei jeder Bekanntmachung oder Aktualisierung zurückgesetzt wird. Überschreitet der Timer einen bestimmten Wert, so führt der Server von sich aus eine Anfrage an die anderen Server im Gebiet aus. Wenn nicht genügend Antworten zurückkommen, wird eine Serverwahl angestoßen.

Tabelle 2-5 stellt die Kommunikations-, Speicher- und Zeitkomplexität von Rendezvous Regions dar. Es werden die gleichen Annahmen wie in 4.2 getroffen. Bei Rendezvous Regions werden Bekanntmachungen von Diensten bei der Installation und Aktualisierungen von Diensten in Abhängigkeit vom jeweiligen Dienst in eine durch die Hashfunktion bestimmte Region des Netzwerks geschickt. Damit ergeben sich Kosten von $O(\sqrt{n})$. Dies gilt ebenso für die Kosten von Suchanfragen und deren Zeitdauer, die proportional zu der Anzahl der benötigten Nachrichtensprünge sind. Weil die verschiedenen Dienste eines Roboters durch die Hashfunktion in unterschiedliche Regionen des Netzwerks abgebildet werden, können die Bekanntmachungen dieser Dienste nicht zusammen versendet werden. Bei m Diensten pro Knoten und n Knoten im Netzwerk ergeben sich zusammen mit den Kosten für die Installation eines Dienstes damit die in der Tabelle angegebenen Kosten für die Gesamtinstallation. Bei Verwendung einer geeignet gewählten Hashfunktion, die die Dienste des Netzwerks gleichmäßig über alle Regionen verteilt, gibt es eine von m abhängige maximale Anzahl von Diensten, die in eine Region abgebildet werden. Diese Anzahl hängt nicht von n ab, wenn die Anzahl der Regionen mit der Anzahl der Knoten steigt. Daraus ergibt sich der in der Tabelle aufgeführte Speicherbedarf der Server in Rendezvous Regions.

Tabelle 2-5. Nachrichten- und Speicherkomplexität sowie Latenz von Rendezvous Regions.

	Installation / Dienst	Gesamtinstallation m Dienste/Knoten	Aktualisierung / Dienst	Suche / Dienst	Latenz / Suche	Speicher / Knoten
RR	$O(\sqrt{n})$	$O(nm\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	O(m)

Eine Stärke von Rendezvous Regions ist seine relativ nachrichten-effiziente Dienstbekanntmachung und Dienstfindung. Durch die Verwendung von Regionen für das Rendezvous (anstatt Punkten, wie beispielsweise beim Vorgänger, dem *Geographic Hash Table*-Ansatz [RBYY+02]), steigt die Robustheit des Systems, da die Genauigkeitsanforderungen an die Positionsinformationen für das Rendezvous gelockert werden.

Die Nachteile resultieren größtenteils aus der Verwendung der geografischen Hashfunktion. Da Dienst-Anfragen mit Hilfe der Hashfunktion auf eine Region abgebildet werden müssen, werden nur semantisch schwache Anfragen niedriger Komplexität unterstützt (Eine Anfrage folgender Art wäre damit beispielsweise nicht möglich: “Wir suchen einen Roboter der 75 Kilogramm bei einer Minimalgeschwindigkeit von 10 km/h transportieren kann. Dabei ist die Geschwindigkeitsanforderung wichtiger und wir könnten eine verminderte Tragfähigkeit bis auf 50 kg akzeptieren, um die gewünschte Geschwindigkeit zu erreichen.“) [CR03, KKO03]. Weiterhin wird das Lokalitätskriterium vernachlässigt. Dienste können auf beliebig weit entfernte Rendezvous Regions abgebildet werden. Dies kann zur Folge haben, dass selbst benachbarte Kunden und Dienstleister eine (in Abhängigkeit von der Größe des Operationsgebiets) beliebig weit entfernte Suchanfrage senden müssen, um einander zu entdecken. Im Fall, dass jeder Knoten mehrere Dienste anbietet, steigt die Zahl der versendeten Nachrichten proportional zu der Anzahl der angebotenen Dienste, da jeder Dienst auf seine eigene Region abgebildet wird. Und mehrere gleichwertige Dienste im Netzwerk steigern nicht unbedingt die Robustheit des Netzwerks, da sie alle auf die gleiche Region abgebildet werden. Im Fall von Netzwerkpartitionen könnten Regionen nicht mehr erreichbar sein, was eine Entdeckung von Diensten verhindert, die dorthin abgebildet wurden, selbst wenn die Dienstleister sich noch im erreichbaren Teil des Netzwerks befinden sollten. Im Fall von Partitionierungen oder leeren Zellen empfehlen die Autoren die Verwendung

mehrerer Hashfunktionen, die Ersatzregionen bereitstellen. Dies würde jedoch den Nachrichtenaufwand wieder erhöhen und somit den größten Vorteil dieses Verfahrens unterlaufen. Schließlich muss für die Lösung in der vorliegenden Form das Operationsgebiet vorher festgelegt werden, es kann nicht dynamisch erweitert oder verschoben werden, was eine wesentliche Einschränkung für mobile Robotersysteme bedeuten kann.

2.2.3 Suchen in internet-basierten Peer-to-Peer-Netzwerken

In den letzten Jahren wurden Peer-to-Peer-Netzwerke zunehmend populärer für den Datenaustausch im Internet [BKKM+03]. Es gibt grundsätzliche Ähnlichkeiten zwischen mobilen Ad-hoc-Netzwerken und Peer-to-Peer-Netzwerken. Beide benötigen a priori kein Wissen über den globalen Zustand des Netzwerks, funktionieren ohne feste Infrastruktur (P2P-Netzwerke nutzen durchaus das Internet als Grundlage, für den Aufbau der P2P-Struktur und für die Organisation der P2P-Teilnehmer wird jedoch keine feste Infrastruktur z. B. in Form dedizierter Server benötigt) und eine Herausforderung in beiden Systemen ist die ständige Ankunft neuer Teilnehmer und die Abmeldung oder der Ausfall bestehender Teilnehmer. Die dezentralisierte Organisation der Netzwerke bietet die Möglichkeit zur spontanen, dynamischen Vernetzung der Teilnehmer und eine im Prinzip hohe Skalierbarkeit des Systems, da jeder Knoten, der sich am Netzwerk beteiligt, auch Speicher, Rechenkapazität und Kommunikationsbandbreite mitbringt. Als Grundproblem müssen in beiden Netzwerken fremde Ressourcen vor der Nutzung gesucht und lokalisiert werden.

Der grundsätzliche Unterschied zwischen beiden Netzwerken besteht darin, dass die Problematik bei mobilen Ad-hoc-Netzwerken auf Netzwerkebene und bei Peer-to-Peer-Netzwerken auf Anwendungsebene betrachtet wird. Die bisher vorgeschlagenen Lösungen für das Entdecken von Ressourcen in P2P-Systemen sehen die logische Kommunikation getrennt von der physikalischen Kommunikation. So verwenden gängige P2P-Lösungen statische Verbindungen auf Anwendungsebene. Für die Kommunikation auf Anwendungsebene wird eine dem Netzwerk überlagerte Struktur aufgebaut und die Erhaltung dieser Struktur wird streng kontrolliert. Das einfache Anwenden eines P2P-Protokolls für das Internet in einem MANET ist aus diesem Grund ineffizient. Das Problem einer solchen Lösung ist, dass Topologie-Änderungen

auf Netzwerkebene - insbesondere bei Mobilität der Knoten - erwartungsgemäß viel häufiger geschehen als auf Anwendungsebene. Dadurch stimmt die von dem P2P-System aufgebaute Verbindungstopologie zwischen den Teilnehmern des Netzwerks schlecht mit der unterliegenden Netzwerktopologie überein. Selbst wenn dies zu Beginn der Fall wäre, würde diese Eigenschaft aufgrund der Mobilität der Netzwerkknoten nach einiger Zeit verschwinden. Die statischen Verbindungen auf Applikationsebene, die auf Netzwerkebene durch Multi-Hop-Verbindungen realisiert wären, müssten aktiv gewartet werden. Um die Verbindungen auf Applikationsebene sicherzustellen, wäre ein nicht unerheblicher Nachrichtenaufwand nötig, um zusammengebrochene Verbindungen zu detektieren und zu reparieren. Weiterhin findet in vielen Protokollen für P2P-Netzwerke die Distanz zwischen Kommunikationspartnern keine Berücksichtigung. So wird auf der logischen Topologie gearbeitet, ohne die physische Distanz der Knoten zu berücksichtigen.

Ein weiterer Grund, warum Lösungen für das Suchen von Ressourcen in internet-basierten P2P-Netzwerken nicht auf mobile Ad-hoc-Netzwerke übertragen werden können, ist die Verwendung von verteilten Hashtabellen. Fortgeschrittene P2P-Protokolle, so genannte strukturierte Ansätze, basieren in der Regel auf verteilten Hashtabellen, die über die Gesamtheit der Knoten verteilt sind. Zu den einzelnen Ressourcen im Netzwerk werden Informationen über den aktuellen Aufenthaltsort dieser Ressourcen durch eine allgemein bekannte Hashfunktion auf bestimmte Knoten des Netzwerks abgebildet. Andere Knoten, die diese Ressource suchen, können über diese Hashfunktion dieselben Knoten berechnen und dann diese über den Aufenthaltsort der Ressource befragen. Der große Vorteil bei der Verwendung solcher verteilten Hashtabellen ist die Möglichkeit, gesuchte Ressourcen in nur O(log(n)) Schritten auf Anwendungsebene zu finden, wenn n die Anzahl der Knoten im P2P-Netzwerk angibt. (Auf Netzwerkebene sind damit im Allgemeinen $O(\log(n)\cdot\sqrt{n})$ Schritte erforderlich, um die Nachrichten an ihr Ziel zu bringen, da die Kommunikationsdistanzen auf Netzwerkebene von den Verfahren oft nicht betrachtet werden und die kommunizierenden Knotenpaare somit beliebig im Operationsgebiet verteilt sind [BKKM+03].) Im Gegensatz dazu benötigen unstrukturierte Ansätze im Mittel oft O(n) Schritte, um eine Ressource garantiert zu finden. Die Verwendung von verteilten Hashtabellen weisen für die Dienst-Entdeckung jedoch einige erhebliche Nachteile auf.

Verteilte Hashtabellen eignen sich gut, wenn es keine Repliken im Netzwerk gibt, sondern viele einzigartige Dienste, die von allen Knoten des Netzwerks auf gleiche

Weise exakt benannt werden können. Dann kann über eine allgemein bekannte Hashfunktion direkt auf den Speicherort mit Informationen über die gesuchte Ressource geschlossen werden. Verteilte Hashtabellen eignen sich weniger, wenn es viele äquivalente Dienste im Netzwerk gibt, Knoten jeweils mehrere Dienste und sich überlappende Mengen von Diensten anbieten, bei 'unscharfen' Anfragen beispielsweise durch Angabe von Platzhaltern oder bei komplexen Anfragen durch Angabe von Parametern [CR03]. Diese Fälle treten aber genau bei der Beschreibung von und der Suche nach Diensten in Roboternetzwerken auf. Jeder Roboter wird vermutlich eine Vielzahl von Diensten anbieten können, allein schon aufgrund der installierten Sensorsysteme. Viele Dienste, wie z. B. Kameras, werden auch relativ oft im Netzwerk verfügbar sein. Bei der Dienstsuche wird es sowohl sehr allgemeine Anfragen (z. B. Sensorinformationen beliebiger Art aus dem Gebiet X) und sehr spezielle Anfragen (z. B. Transportroboter mit mindestens Y kg Nutzlast und einer Transportgeschwindigkeit von Z) geben. Diese Fälle sind mit verteilten Hashtabellen nur aufwendig zu realisieren und resultieren in einem hohen Nachrichtenaufkommen.

Die direkte Anwendung von P2P-Protokollen für das Internet in einem mobilen Ad-hoc-Netzwerk ist daher nicht sinnvoll [GLR05, DB04].

Als Hinweis sei noch angemerkt, dass der in (2.2.2.2) vorgestellte Grid Location Service ein Beispiel für einen Ansatz ist, der auf einer verteilten Hashtabelle beruht und speziell für mobile Ad-hoc-Netzwerke entworfen wurde. Bei dieser Lösung wird die zugrunde liegende Netzwerktopologie berücksichtigt. Allerdings eignet sich GLS durch die Verwendung der Hashfunktion wie oben erklärt dennoch nicht gut für die Entdeckung von Diensten. Entsprechend ist GLS von seinen Entwicklern auch für die Lokalisierung von – im Netzwerk jeweils einzigartigen – Knoten konzipiert. Dadurch wirkt sich die Verwendung der Hashfunktion nicht aus.

2.3 Roboternetzwerke und Robotersysteme mit serviceorientierten Architekturen

Trotz des zunehmenden Interesses an Multi-Roboter-Systemen befindet sich die Forschung an Kommunikationssystemen für derartige Systeme noch in ihren Anfängen. Wenige Roboternetzwerke wurden bisher realisiert. Ein Beispiel ist das *Centibots*-Projekt [KFOA+04] bei dem ein Netzwerk aus 100 Robotern aufgebaut wurde (Abb. 2-11). Im Rahmen des Projekts wurde ein auf Jini basierendes Framework entwickelt, mit dem die Roboter verwaltet werden können. Die einzelnen Roboter werden dabei als Dienste betrachtet. Dies ist eines der ersten Projekte, das den Nutzen von Lösungen für die Dienst-Entdeckung in großen Roboter-Netzwerken verdeutlicht. Allerdings weist Jinis Entdeckungsmechanismus Schwächen bei der Anwendung in Roboternetzwerken auf (siehe 2.2.1.2). Die in dieser Dissertation entwickelte Lösung beseitigt diese Schwächen.

Abbildung 2-11. Das Centibots-Projekt. Eines der bisher größten realisierten Roboternetzwerke mit 100 Robotern. (Bild: [SRI])

Ein weiteres Beispiel für ein großes Roboternetzwerk findet sich in [HPS06]. Hier hat ein heterogenes Team aus etwa 80 Robotern die Aufgabe, ein unbekanntes Stockwerk in einem Gebäude zu erkunden, zu kartografieren und sich über das Operationsgebiet zu verteilen. Nach der Stationierung der Roboter sollen die Roboter mögliche Eindringlinge detektieren und verfolgen. Die dabei gesammelten Daten werden über das Kommunikationsnetzwerk an einen aus der Ferne operierenden menschlichen Bediener gesendet. In ihrer Arbeit benennen die Autoren die Kommunikationsinfrastruktur als einen der kritischen, noch nicht zufrieden stellenden Punkte des Projekts.

In [LMP03] wird Dienst-Entdeckung genutzt, um Sensorfehler in einem Multi-Roboter-System zu diagnostizieren und zu beheben. Jini wird verwendet, um Dienste der Roboter wie die Bereitstellung von Sensordaten oder Verhaltensmuster zu entdecken. In diesem Artikel beschäftigen sich die Autoren allerdings hauptsächlich mit dem Problem, wie fehlerhafte und ausgefallene Sensoren detektiert und kompensiert werden können. Das betrachtete Netzwerk ist nur von kleiner Größe und die Kommunikation basiert auf Broadcasts.

In [LGMV05] wird die *Distributed Field Robot Architecture* präsentiert. DFRA ist eine Architektur, in der alle Fähigkeiten eines Roboters als Dienste implementiert werden, einschließlich Sensoren, Aktoren und Verhalten. Die Dienste sind nicht nur dem jeweiligen Roboter selbst zugänglich, sondern werden über ein Jini-basiertes Framework exportiert. Auf diese Weise werden die Dienste auch anderen Robotern und menschlichen Benutzern zugänglich gemacht. Die Fähigkeiten jedes Roboters können dynamisch während der Laufzeit entdeckt werden und Dienste können anhand ihrer Funktionalität gesucht werden. Auch hier ist das betrachtete Netzwerk nur von kleiner Größe und die Kommunikation basiert auf Broadcasts.

Die Integration von mobilen Robotern in “intelligente“ (mit Rechen- und Speicherkapazitäten oder Sensoren ausgestattete) Umgebungen wird in [HSC05] betrachtet. Roboter können in verschiedene Zimmer geschickt werden und bedienen dort in der Umgebung eingebettete Geräte wie beispielsweise Lichtschalter oder Fensterblenden. Die Software basiert auf Web-Services, allerdings sind die Schnittstellen (im Gegensatz zu der Lösung in 6.4) nicht in den Roboter selbst integriert. Stattdessen erhalten die Roboter ihre Anweisungen von einem externen Rechner. Auch die Entdeckung der in der Umgebung des Roboters vorhandenen Ressourcen erfolgt nicht dynamisch. Die Autoren verwenden so genannte *Knowledge Bases*, die schon im Voraus Informationen über alle in einem Raum verfügbaren Ressourcen besitzen.

Microsoft Robotics Studio [Micr06] stellt eine serviceorientierte Architektur zur Programmierung von Robotern bereit. Es besteht eine hohe Ähnlichkeit zu Web-Services, bei der Konzeption wurde auf Einfachheit, Interoperabilität und eine lose Kopplung der Dienste und Anwendungen Wert gelegt. Anwendungen werden kreiert durch die Komposition von lose gekoppelten Diensten, die sich sowohl auf einem einzelnen Rechnersystem befinden oder aber auch über das Netzwerk verteilt sein können. Ebenso können einfache Anwendungen zu komplexeren kombiniert werden. Durch die so genannten *Decentralized System Services* (DSS) werden grundlegende Dienste zur Unterstützung des Benutzers bzw. des Systems zur Verfügung gestellt. Dies betrifft Aufgaben wie das Debugging, das Beobachten und Aufzeichnen von Laufzeitdaten, das Einhalten grundsätzlicher Sicherheitsaspekte, die Datenspeicherung und schließlich auch die Dienst-Entdeckung. Die Dienst-Entdeckung basiert allerdings auf einem bekannten, zentralen Verzeichnis, bei dem sich Dienstleister registrieren und das Kunden nach geeigneten Diensten befragen können. Dieses Verfahren eignet sich nur bedingt für große, drahtlose Multi-Hop-Netzwerke und weist die gleichen Schwächen auf wie etwa UDDI (2.2.1.3) oder Jini (2.2.1.2). Genau für diese Problematik bietet das in dieser Arbeit entwickelte Protokoll zur Dienst-Entdeckung in Roboternetzwerken eine Lösung.

3. Servicebasierte Multi-Roboter-Systeme

In diesem Kapitel wird die Grundidee servicebasierter Multi-Roboter-Systeme vorgestellt. Im ersten Teil wird auf eine vergleichbare Entwicklung in der Softwareindustrie hingewiesen und die Vorteile servicebasierter Multi-Roboter-Systeme erläutert. Dann wird der Begriff der Dienstleistung spezifiziert und der Vorgang der Dienst-Entdeckung genauer betrachtet. Es werden beispielhaft einige denkbare Dienste aus der Robotik aufgelistet. Schließlich folgt noch ein Vergleich des servicebasierten Ansatzes mit anderen Entwurfsstrategien für Multi-Roboter-Systeme.

3.1 Die Grundidee servicebasierter Multi-Roboter-Systeme

In der Softwareindustrie lässt sich eine Entwicklung von lokalen Anwendungen über erste verteilte Anwendungen hin zu den zurzeit propagierten serviceorientierten Architekturen beobachten [SH04b]. Anfangs wurden Programme lokal auf Einzel-Rechner-Systemen oder über das Netzwerk auf zentralisierten Client-Server-Systemen ausgeführt. Die ersten verteilten Systeme erlaubten es, Programme über das Netzwerk zu verteilen - sie mussten nicht mehr in einem einzigen Rechner ausgeführt werden. Programmteile konnten nun ihre Funktionalität im Netzwerk bekannt machen und genutzt werden. Durch die Vereinbarung von Schnittstellen konnten die einzelnen Programmteile außerdem in verschiedenen Programmiersprachen implementiert und auf verschiedenen Plattformen installiert sein. Trotz der Verteiltheit waren derartige Anwendungen jedoch meist immer noch monolithische Einheiten eng gekoppelter Programmfragmente mit oft handcodierten Aufrufen bekannter Funktionen. In den letzten Jahren vollzog sich evolutionär der Schritt zu serviceorientierten Architekturen. In diesem Ansatz bieten sich autonome Teilnehmer des Netzwerks gegenseitig Dienste an. Wenn eine Funktionalität benötigt wird, über die man selbst nicht verfügt, wird im ersten Schritt ein geeigneter Dienstleister lokalisiert. In diesem Modell wird eine Anwendung als serielle oder parallele Abarbeitung einer Folge von Dienstleistungen betrachtet. Durch diese Art der Organisation kann das Gesamtsystem flexibler in der Aufgabenbearbeitung und robuster gegen den Ausfall einzelner Teile werden. Dienste können dynamisch hinzugefügt werden, um die Fähigkeiten des Gesamtsystems zu

erweitern. Dabei muss sich im Fall einer Selbstorganisation des Systems die Entwurfskomplexität des Systems nicht zwangsläufig erhöhen.
Ein weiterer Trend in der Informationstechnologie wird zurzeit unter anderem unter den Begriffen *Pervasive* bzw. *Ubiquituous Computing* vorangetrieben und beschreibt die Vernetzung einer extrem hohen Anzahl von einfachen Teilnehmern. Dieser Ansatz ist aus Sicht der Robotik am ehesten zu vergleichen mit den Ansätzen der Schwarmrobotik (siehe auch 3.4).

In der Robotik lässt sich eine vergleichbare Evolution von Einzel-Roboter-Systemen über eng gekoppelte, kleine Multi-Roboter-Systeme bis hin zu lose gekoppelten, großen Multi-Roboter-Systemen feststellen – auch wenn die Entwicklung noch nicht so weit fortgeschritten wie in der Softwareindustrie ist. Jede dieser Systemarten erfordert eigene Entwurfsstrategien. Der servicebasierte Ansatz stellt eine Möglichkeit dar, um große Multi-Roboter-Systeme zu beherrschen. Dabei tauschen spezialisierte, lose gekoppelte Roboter Dienstleistungen miteinander aus. Solch ein Austausch von Dienstleistungen ist eine mögliche Form, um die Zusammenarbeit einer großen Anzahl von individuellen Robotern zu organisieren und zu beschreiben. Die Dienst-Entdeckung, das Suchen und Finden eines geeigneten Dienstes in einem Netzwerk aus vielen Dienstanbietern, bildet hierfür die Grundlage.

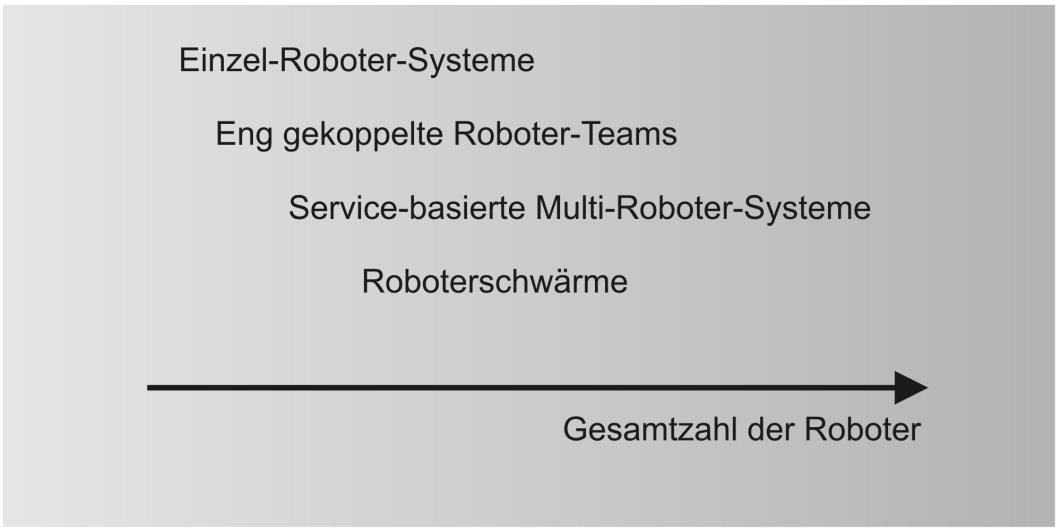

Abbildung 3-1. Verschiedene Ansätze für den Entwurf von (Multi-) Roboter-Systemen in Abhängigkeit von der Anzahl der Roboter.

In einer servicebasierten Architektur können Roboter oder menschliche Nutzer geeignete Roboter für bestimmte Aufgaben bei Bedarf aus der Umgebung rekrutieren. Durch die Verwendung offener, standardisierter Schnittstellen wird eine Entkopplung von Implementierung und Funktionalität vorgenommen und Interoperabilität sichergestellt, so dass die Funktionalität eines Roboters von jedem anderen Roboter genutzt werden kann. Die Roboter sind als Dienstleister völlig autonome Einheiten, die selbst entscheiden, für welche Aufgaben sie geeignet sind, und die Aufgaben selbstständig durchführen. So werden eine dynamische Zusammenarbeit und eine flexible Rekombination der individuellen Roboter möglich. Verschiedene Robotertypen und Plattformen unterschiedlicher Hersteller können interagieren und gegeneinander ausgetauscht werden. Damit ist es auch möglich, die Implementierung einzelner Roboter zu ändern, ohne gleich Auswirkungen auf das Gesamtsystem befürchten zu müssen. Roboter können einfach ersetzt, hinzugefügt und entfernt werden. Servicebasierte Multi-Roboter-Systeme, basierend auf effizienten Lösungen für die Dienst-Entdeckung, haben das Potential, die Entwicklung und Nutzung großer Multi-Roboter-Systeme stark zu vereinfachen.
Der servicebasierte Ansatz eignet sich speziell für Szenarien, in denen große Anzahlen von heterogenen Robotern in einem Operationsgebiet aktiv sind. Ein Beispiel sind Rettungsmaßnahmen im großen Umfang, in denen Hunderte oder sogar Tausende von Robotern mobilisiert werden können [KTNM+99]. Eines Tages könnten mobile, autonome Roboter auch Teil unserer täglichen Umwelt werden und in großer Anzahl in unseren Büros, Fabriken und Haushalten installiert sein.

3.2 Dienst-Entdeckung

Eine Dienstleistung kann als ein Prozess betrachtet werden, der einen Nutzen für den Kunden erzeugt durch eine Änderung am Kunden selbst oder seiner physischen oder immateriellen Güter [CS06]. Die Ausführung einer Dienstleistung umfasst den Dienstleister, den Kunden und die Ausrüstung, die dazu verwendet wird, um die Dienstleistung zu erbringen.

In dieser Arbeit wird dabei folgende Terminologie verwendet. Ein **Dienst** führt Funktionen für den Nutzer dieses Dienstes aus. Dieser Vorgang wird auch als das

Erbringen einer **Dienstleistung** bezeichnet. Eine Einheit, die einen oder auch mehrere solcher Dienste zur Verfügung stellt, wird als **Dienstleister** oder auch Dienstanbieter bezeichnet. Die Einheit, für den der Dienst die Funktion ausführt, ist der **Dienstnutzer** oder Kunde. Die Lokalisierung eines Dienstes und des zugehörigen Dienstleisters ist die **Dienst-Entdeckung**.

Der Austausch von Dienstleistungen ist eine mögliche Form, um die Zusammenarbeit von Individuen zu organisieren und zu beschreiben. Jedes Individuum stellt seine Fähigkeiten in Form von Diensten zur Verfügung, die von anderen ausgewählt und genutzt werden können. Die Dienst-Entdeckung, das Suchen und Finden eines geeigneten Dienstes in einem Netzwerk aus vielen Dienstanbietern, bildet hierfür die Grundlage.

Bei einer Dienst-Entdeckung in technischen Systemen sind, vergleichbar zu nicht-technischen Systemen, drei Typen von Akteuren involviert. Kunden, die Funktionalitäten suchen und in Anspruch nehmen wollen, Dienstleister, die diese Funktionalitäten erbringen, und optional Verzeichnisse von Diensten, die als Vermittler zwischen Kunden und Dienstleistern fungieren. Verzeichnisse können natürlich selbst auch Kunden sein oder andere Dienstleistungen anbieten.
Es folgt ein einfaches Beispiel, wie eine Dienst-Entdeckung ablaufen kann. Ein Kunde benötigt eine Dienstleistung. Er weiß nicht, ob es überhaupt ein Gerät gibt, das den gesuchten Dienst im erreichbaren Teil des Netzwerks anbietet und falls ja, wo sich dieses Gerät befindet, weder geographisch, noch bezüglich der Netzwerktopologie. Im einfachsten Fall würde der Kunde eine Suchanfrage durch Fluten des Netzwerks durchführen und auf eine Antwort eines geeigneten Dienstleisters warten. Ein Ausbleiben einer Antwort würde, eventuell nach mehreren Versuchen, als das Fehlen des gewünschten Dienstes im erreichbaren Teil des Netzwerks interpretiert. Für größere Netzwerke sind Verzeichnisse eine sinnvolle Ergänzung. Sie sammeln aktiv oder auch passiv Informationen über die im Netzwerk verfügbaren Dienstleister und die von ihnen angebotenen Dienste. Der Kunde kann seine Anfrage nun direkt an einen oder auch mehrere solcher Verzeichnisse senden, soweit sie ihm bekannt sind. Nachdem ein geeigneter Dienstleister gefunden ist, kann der Kunde dessen Funktionalität in Anspruch nehmen.

Die grundsätzlichen Aktivitäten der Dienst-Entdeckung beinhalten

- optional: Die Bekanntgabe von Diensten und ihrer Eigenschaften. Dadurch kann das Finden von Diensten erleichtert werden.
- das Finden von Diensten innerhalb der Netzwerktopologie (oder geografisch bei physischen Diensten).
- und optional: Die Nutzung der Dienste nach ihrer Entdeckung. Manche Protokolle zur Dienst-Entdeckung unterstützen den Dienstnutzer bei diesem Schritt (wie beispielsweise im Fall von Jini durch die Bereitstellung von Java-Objekten, mit denen auf die Dienste zugegriffen werden kann), andere überlassen es dem Dienstanbieter und dem Dienstnutzer, sich nach der Entdeckung selbst über das weitere Vorgehen zu einigen (wie bei Bluetooth).

Eine Lösung für die Dienst-Entdeckung sollte dabei folgende Grundeigenschaften erfüllen [AK05]:

- Sie muss Dienste im Netzwerk entdeckbar machen und das automatisierte Entdecken von Diensten ermöglichen.
- Sie sollte es Diensten ermöglichen, ihre angebotene Funktionalität zu beschreiben, und diese Beschreibung an potentielle Dienstnutzer weiterleiten.
- Sie sollte interoperabel sein, also eine Bandbreite von unterschiedlichen Gerätetypen mit unterschiedlicher Ausstattung an Speicher, Rechenkapazität oder Betriebssystem unterstützen. Dabei sollte diese Heterogenität dem potentiellen Nutzer des Dienstes verborgen bleiben, es sei denn, sie ist für die Auswahl eines Dienstes von Bedeutung.
- Die Lösung sollte flexible Anfragen ermöglichen. Es sollte sowohl die Suche nach sehr spezifischen Diensten unterstützt werden, wie auch sehr allgemeine Suchen nach oft verfügbaren Diensten oder Anfragen nach einer Übersicht von Diensten mit bestimmten Grundfähigkeiten, um dann eine Auswahl zu treffen.
- Sie sollte für Dienstanbieter wie auch Dienstnutzer sicher sein und sie vor bösartigen Diensten wie auch Kunden schützten. Autorisierungs- und Authentifizierungsverfahren stellen beispielsweise sicher, dass nur berechtigte Geräte Zugriff auf die angebotenen Dienste haben und dass die Teilnehmer des Netzwerks tatsächlich mit den gedachten Partnern kommunizieren.
- Die Lösung sollte effizient und skalierbar sein und in Umgebungen mit hoher, aber auch in Umgebungen mit niedriger Gerätedichte funktionieren.

- Sie sollte für mobile, dynamische Umgebungen geeignet sein. Das Auftauchen oder der Wegfall von Diensten und Zustandsänderungen sollten möglichst zeitnah widergespiegelt werden.

Die beiden letzten Punkte sind allgemeingültig, haben aber gerade für mobile, drahtlose Netzwerke eine besondere Bedeutung.

Formalisiert lässt sich das Problem der Dienst-Entdeckung folgendermaßen beschreiben. Es wird ein Zeitraum

$$T = [t_A, t_E], \qquad t_E > t_A, t_A \in \mathbb{R}, t_E \in \mathbb{R} \qquad (3.1)$$

betrachtet.

$$N(t) = \{k_{ID1}, k_{ID2}, ..., k_{IDn(t)}\}, \qquad n(t) \in \mathbb{N}^+ \qquad (3.2)$$

bezeichne die Menge der Knoten im Netzwerk zum Zeitpunkt $t \in T$. Jeder Knoten k_{IDi} ($1 \leq i \leq n(t)$) bietet zum Zeitpunkt $t \in T$ die Menge

$$D_{IDi}(t) = \{d_{IDi,1}, d_{IDi,2}, ..., d_{IDi,p(t)}\}, \qquad p(t) \in \mathbb{N} \qquad (3.3)$$

von Diensten an. Zu den Zeitpunkten

$$T_s = \{t_{s1}, t_{s2}, ..., t_{sq}\} \subset T, \qquad q \in \mathbb{N}^+ \qquad (3.4)$$

werden von jeweils einem Knoten aus den Mengen $N(t_{s1})$, $N(t_{s2})$, … bzw. $N(t_{sq})$ Dienstsuchen nach den Diensten d_{s1}, d_{s2}, ... bzw. d_{sq} gestartet. Dabei wird versucht, innerhalb des Zeitraums

$$T_{sj} = [t_{sj}, t_{sj} + \Delta], \qquad 1 \leq j \leq q, \Delta \in \mathbb{R}^+ \qquad (3.5)$$

eine Menge

$$N_{sj} = \{k_{sj,ID1}, k_{sj,ID2}, ..., k_{sj,IDr}\}, \qquad r \in \mathbb{N} \qquad (3.6)$$

von Knoten im Netzwerk zu finden, die den gesuchten Dienst d_{sj} bereitstellen. Wenn $D_{sj,IDu}$ ($0 \leq u \leq r$) die Menge der Dienste von $k_{sj,IDu}$ im Zeitraum T_{sj} bezeichnet, muss also gelten

$$d_{sj} \in D_{sj,ID1}, d_{sj} \in D_{sj,ID2}, ..., d_{sj} \in D_{sj,IDr} \tag{3.7}$$

Für diesen Prozess der Dienst-Entdeckung sind unterschiedliche Optimierungsziele denkbar. Beispiele sind die Minimierung der Latenz bis zum Eintreffen der ersten Suchantwort oder die Minimierung des Nachrichtenaufkommens. Es bezeichne $t_{sj,a1}$ den Zeitpunkt, zu dem der erste Knoten $k_{sj,ID1}$ gefunden wird, der den Dienst d_{sj} anbietet. Es sei w_{sj} ein Maß für die Nachrichtenmenge, die bei der Suche nach Dienst d_{sj} versendet wird. Dann stellen sich die Optimierungsziele dar als

$$\text{minimiere } \max_{j=1}^{q}\left(t_{sj,a1} - t_{sj}\right) \quad \text{bzw.} \quad \text{minimiere } \max_{j=1}^{q}\left(w_{sj}\right), \tag{3.8}$$

wenn die maximale Latenz bzw. Nachrichtenmenge minimiert werden soll, oder

$$\text{minimiere } \sum_{j=1}^{q}\left(t_{sj,a1} - t_{sj}\right) \quad \text{bzw.} \quad \text{minimiere } \sum_{j=1}^{q} w_{sj}, \tag{3.9}$$

wenn die Summe der Latenzen bzw. der Nachrichtenmengen minimiert werden soll.

In [Helm05] wird eine mögliche Kategorisierung von Anfragen in Kommunikationsnetzwerken vorgestellt (Tabelle 3-1). Nach dem Schema können Suchanfragen danach unterschieden werden, ob die Antwortdaten zusammengefasst werden können (zusammenfassbar/nicht zusammenfassbar), ob die Daten im Netzwerk einzigartig sind oder mehrfach zur Verfügung stehen (einzigartig/nicht einzigartig), ob die Daten kontinuierlich oder nur einmal benötigt werden (kontinuierlich/einmalig) und ob die Anfragen selbst einfach oder komplex aufgebaut sind (einfach/komplex). Anfragen zur Dienst-Entdeckung fallen in der Regel in die Kategorien nicht zusammenfassbar, nicht einzigartig, einmalig und komplex.

Tabelle 3-1. Eine mögliche Kategorisierung von Suchanfragen. Suchanfragen zur Dienst-Entdeckung im Speziellen fallen hier in die Kategorien nicht zusammenfassbar, nicht einzigartig, einmalig und komplex (grau unterlegt).

Können Antwortdaten zusammengefasst werden (z.B. Durchschnittstemperatur verschiedener Sensoren eines Gebiets) oder nicht?	Zusammenfassbar	Nicht zusammenfassbar
Sind die gesuchten Daten einzigartig oder können sie mehrfach im Netzwerk vorkommen?	Einzigartig	Nicht einzigartig
Werden die Daten kontinuierlich benötigt (z. B. Temperaturmessreihe) oder nur einmalig?	Kontinuierlich	Einmalig
Sind Anfragen einfach oder komplex (z. B. durch Angabe von Parametern, Platzhaltern, Booleschen Verknüpfungen) aufgebaut?	Einfach	Komplex

3.3 Mögliche Dienstleistungen in der Robotik

Im Fall von mobilen Robotern ist anzunehmen, dass der Austausch physischer Dienstleistungen im Vordergrund stehen wird. Die Interaktionen müssen jedoch keineswegs darauf beschränkt bleiben. Tabelle 3-2 zeigt eine Reihe von Dienstleistungen im Bereich der Robotik und deren Aufschlüsselung bezüglich ihrer Positionsabhängigkeit, d.h. ob Dienstleister und Kunde an spezifischen geografischen Positionen sein müssen, um Dienste zu erbringen oder in Anspruch zu nehmen.

Tabelle 3-2. Mögliche Dienste in der Robotik aufgeschlüsselt nach ihrer Positionsabhängigkeit.

	Dienstleistung
Dienstleister/Kunde muss an einer bestimmten Position sein	Physische Manipulation (z. B. Transport, Herstellung oder Reparatur materieller Objekte) Auf-/Umrüstung von Robotern mit Erweiterungsmodulen und ggf. den entsprechenden Fähigkeiten für deren Nutzung

	Ladestationen für Roboter Bereitstellung von Sensordaten durch Roboter Kontrolle von Robotern durch Software-Agenten (Roboter stellen temporär Software-Agenten ihre „Körper“ zur Verfügung, damit diese Aufgaben bearbeiten können.) …
Dienstleister/Kunde muss nicht an einer bestimmten Position sein	Auslagerung von Daten, Berechnungen durch Roboter Herunterladen neuer Fähigkeiten durch Roboter …

Im Fall von physischen zu erbringenden Dienstleistungen stehen physische Manipulationen der Umwelt im Vordergrund. Bei derartigen Dienstleistungen müssen Dienstanbieter und/oder Kunde am geeigneten Ort sein, um die Dienstleistung auszuführen oder in Anspruch nehmen zu können. Ein typisches Beispiel hierfür ist der Materialtransport. Die Nutzung von Ladestationen durch Roboter ist ein weiteres Beispiel, bei dem die geografische Position des Dienstleisters von erheblicher Bedeutung für seine Auswahl ist. Ein Roboter, der eine Ladestation benötigt, wird versuchen, möglichst eine in seiner direkten Umgebung zu lokalisieren.

Im Gegensatz hierzu sind auch Dienstleistungen denkbar, bei denen der Informationsaustausch im Vordergrund steht. Dienstleistungen informationstechnischer Art können sowohl positionsabhängig als auch primär positionsunabhängig sein. Ein Beispiel für eine positionsabhängige Dienstleistung ist die Bereitstellung von Sensordaten durch einen Roboter. Ein Kunde, der die Bereitstellung von Sensordaten wünscht, wird versuchen einen Roboter zu lokalisieren, der sich bereits in der Nähe derjenigen Position befindet, an der die Sensordaten anfallen. Ein Beispiel für eine positionsunabhängige Dienstleistung ist das Herunterladen neuer Fähigkeiten durch einen Roboter. Hier ist die Diensterbringung erst einmal unabhängig von den direkten Aufenthaltsorten von Dienstleister und Kunde. Die Positionen der Kooperationspartner fließen durchaus indirekt ein, weil beispielsweise versucht wird, die Kommunikationsdistanz zwischen den Kooperationspartnern zu minimieren. Aber die

genauen geografischen Positionen sind dabei weniger von Interesse als die Entfernung innerhalb der Kommunikationsinfrastruktur. Dieser Unterschied tritt dabei besonders deutlich zutage, wenn Barrieren die Bewegung des Roboters zum Ort der Diensterbringung verhindern. Dies hat erhebliche Auswirkungen auf die Auswahl der Dienstleister in den positionsabhängigen Fällen, braucht aber in den positionsunabhängigen Fällen nicht berücksichtigt zu werden. Umgekehrt haben Barrieren, die den Aufbau einer stabilen Kommunikationsverbindung verhindern, eher Einfluss auf die positionsunabhängigen Dienste.

Die Hauptaufgaben mobiler Roboter werden vermutlich tendenziell eher in die Kategorie der positionsabhängigen Dienste fallen, da sie hier ihre Stärken, ihre Mobilität und die Fähigkeit zur physischen Manipulation ihrer Umwelt, ausspielen können.

3.4 Servicebasierte Multi-Roboter-Systeme, Roboter-Teams und Schwarmintelligenz

Der servicebasierte Ansatz ist eine zusätzliche, ergänzende Sicht auf den Entwurf von Multi-Roboter-Systemen, der sich zwischen schwarmbasierten Ansätzen auf der einen Seite und eng gekoppelten, spezialisierten Roboter-Teams auf der anderen Seite positioniert.

Schwarmbasierte Ansätze [BDT99] sind charakterisiert durch einfache Agenten, die einer kleinen Menge von simplen Verhaltensregeln gehorchen. Kommunikation, entweder explizit oder implizit über die Umwelt, ist strikt lokal begrenzt. Es gibt keine zentrale Koordination, Kooperation zwischen den Agenten und globale Effekte bilden sich als Ergebnis der individuellen Verhaltensregeln heraus. Schwarmsysteme bestehen in der Regel aus sehr vielen homogenen Agenten und verfügen über eine hohe Redundanz.

In direktem Gegensatz zu schwarmbasierten Systemen stehen eng gekoppelte, kleine Teams spezialisierter, oft heterogenerer Roboter, die zur Lösung spezifischer Aufgaben entwickelt wurden, wie beispielsweise in [WGDK+02]. Die einzelnen Roboter sind

typischerweise in der Lage, Teilaufgaben selbstständig zu erledigen. Durch explizite Koordination und Kooperation löst das Team Aufgaben, die jenseits der Fähigkeit der einzelnen Teammitglieder sind. Mögliche Formen der Kooperation und die Zeitpunkte, wann sie erfolgen, werden in der Regel im Voraus durch den Systementwickler festgelegt. Die Teammitglieder kennen einander, ihre jeweiligen Fähigkeiten und wissen, wie sie miteinander kommunizieren und interagieren können, oft basierend auf einem proprietären Protokoll. Da die Teams klein sind, befinden sich die Teammitglieder im Allgemeinen innerhalb der Kommunikationsreichweite der anderen und die Kommunikation erfolgt über Broadcasts.

Im Unterschied zu den obigen Ansätzen beruht der servicebasierte Ansatz auf dem Austausch von Dienstleistungen zwischen lose gekoppelten Agenten. Jeder Roboter wird auf bestimmte Funktionen spezialisiert sein, die er selbstständig ausführen und anderen Robotern als Dienstleistung anbieten kann. Wenn ein Roboter eine Funktionalität benötigt, über die er selbst nicht verfügt, rekrutiert der Roboter andere geeignete Roboter aus seiner Umwelt, um diese Aufgabe auszuführen. Kooperation und das Ergebnis der Operationen sind intentional und nicht emergent wie im schwarmbasierten Ansatz. Zeit, Ort und Art der Kooperation erwachsen jedoch aus den Umständen. Es ist vorher nicht bekannt, wann welche Art von Kooperation notwendig wird. Um Interoperabilität sicherzustellen, basiert die Kommunikation auf offenen, standardisierten Schnittstellen, so dass die Funktionalitäten eines Roboters durch jeden anderen Roboter nachgefragt werden können. Die Kommunikation wird aus Gründen der Skalierbarkeit meistens lokal stattfinden, dies ist jedoch kein Paradigma wie im schwarmbasierten Ansatz. Gelegentlich wird es notwendig sein, auch über weite Entfernungen zu kommunizieren, um einen geeigneten Dienstleister zu finden. Es ist wahrscheinlich, dass es Redundanz im System gibt, da die Roboter vor allem in größeren Roboternetzwerken sich überlappende Mengen von Diensten anbieten werden. Ein illustratives Beispiel für den servicebasierten Ansatz ist ein Roboter, der einen blockierten Weg entlang fahren will. Durch Dienst-Entdeckung lokalisiert und ruft er einen Transportroboter aus seiner Umgebung, der das Hindernis entfernt, so dass er weiterfahren kann. Der Transportroboter muss keineswegs darauf spezialisiert sein, Hindernisse für andere Roboter zu entfernen. Und die beiden Roboter brauchen keine sonstige Verbindung zueinander zu haben, weder vor diesem Ereignis noch danach. Der

herbeigerufene Transportroboter war einfach ein zu dem Zeitpunkt und an jenem Ort der geeignete Roboter, um diese Aufgabe zu erfüllen.

In Tabelle 3-3 werden die drei Ansätze noch einmal nach verschiedenen Unterscheidungsmerkmalen aufgeschlüsselt und verglichen.

Tabelle 3-3. Verschiedene Ansätze für den Entwurf von Multi-Roboter-Systemen im Vergleich.

	Teambasiert	**Servicebasiert**	**Schwarmbasiert**
Anzahl der Roboter	Klein	Klein bis groß	Groß
Komplexität der Individuen	Hoch	Hoch	Gering
Homogenität	Eher heterogen	Heterogen mit Repliken	Homogen
Kommunikationsreichweite	Lokal	Lokal, ggf. global	Lokal
Kommunikationsprotokolle	Proprietär	Standardisiert	Implizit über Umwelt / explizit proprietär
Koordination	Zentral	Zentral / dezentral	Dezentral
Globale Effekte, Erreichen des Ziels	Intentional	Intentional/ Emergent	Emergent

4. Zellenbasierte Dienst-Entdeckung (CSD: Cell-based Service Discovery)

In diesem Kapitel wird das *Cell-based Service Discovery - Protokoll (CSD)* vorgestellt, das in Rahmen dieser Arbeit für die Dienst-Entdeckung in Roboternetzwerken entwickelt wurde. Zu Beginn werden die Anforderungen an und die Vorraussetzungen für CSD aus netzwerktechnischer Sicht formuliert. Danach wird das Protokoll erst in einer Übersicht und dann im Detail mit möglichen Erweiterungen vorgestellt. Weiterhin werden spezielle Fehlerszenarien, wie beispielsweise das Vorhandensein von Hindernissen, und deren Auswirkungen auf CSD betrachtet.
Im zweiten Teil wird CSD analytisch untersucht und mit den anderen in 2.2.2 vorgestellten Verfahren für die Dienst-Entdeckung in mobilen Ad-hoc-Netzwerken verglichen. Ebenso werden die Auswirkungen der Existenz von Repliken, also mehrerer gleichwertiger Exemplare von Diensten im Netzwerk, auf CSD betrachtet. Abgeschlossen wird dieses Kapitel mit Empfehlungen für die Einsatzszenarien von CSD.

4.1 Anforderungen und Voraussetzungen

In mobilen Ad-hoc-Netzwerken haben Knoten die Eigenschaft der physischen Mobilität. Daher kann die Netzwerktopologie sehr dynamisch und instabil sein. Dies gilt insbesondere für Roboternetzwerke, weil die Erbringung physischer, örtlich verteilter Aufgaben die Hauptaufgabe mobiler Roboter ist und sie daher regelmäßig in Bewegung sein werden. Dennoch sollten Kommunikationsprotokolle für derartige Netzwerke auch für den schlechtmöglichsten Fall geeignet sein – wenn sich alle Knoten ständig bewegen. Die physische Aufgabenbearbeitung wird außerdem tendenziell eine verringerte Anfragerate nach Diensten zur Folge haben, weil die Ausführung physischer Aufträge im Unterschied beispielsweise zu digitalen Interaktionen in Computernetzwerken tendenziell längern dauern wird. Zusammen mit der Mobilität der Knoten ergibt sich damit ein hohes Verhältnis von Dienstbekanntmachungen zu Dienstsuchen in Roboternetzwerken. In Roboternetzwerken und mobilen Ad-hoc-Netzwerken im Allgemeinen sind weiterhin die Kommunikationsverbindungen drahtlos.

Um größere Entfernungen zu überbrücken, werden Multi-Hop-Verbindungen genutzt. Dies hat verschiedene Gründe, wie die beschränkten Sendereichweiten der Knoten, das Ziel Energie zu sparen, da die benötigte Sendeleistung mindestens quadratisch mit der Entfernung zunimmt, oder die Vermeidung von Interferenzen beim Senden.

Neben den Kriterien aus 3.2, durch die Anforderungen an Lösungen zur Dienst-Entdeckung im Allgemeinen beschrieben werden, werden daher aus netzwerktechnischer Sicht noch folgende weitere Anforderungen an die Lösung gestellt:

- **Robustheit:** Ein Netzwerk aus mobilen Robotern kann sehr dynamisch sein. Knoten werden ausfallen und den Empfangsradius anderer Knoten oder das Netzwerk verlassen. Neue Knoten werden sich dem Netzwerk anschließen. Das Fehlen einzelner Knoten sollte nur Auswirkungen auf kleine Teile des Netzwerks haben und neue Knoten sollten schnell integriert werden.
- **Lokalität:** Kommunikation über Multi-Hop-Verbindungen ist aufwendig, langsam und aufgrund der Dynamik der Netzwerktopologie unzuverlässig. Die Lösung sollte, insbesondere bei Dienstbekanntmachungen, möglichst nur auf lokale Kommunikationsverbindungen zurückgreifen. Ein weiterer Vorteil bei weitgehender Verwendung von lokalen Kommunikationsverbindungen ist eine Erhöhung der Robustheit des Gesamtsystems, da bei Netzwerk-Partitionierungen die einzelnen Teile des Netzwerks für sich jeweils ohne große Restrukturierungen funktionsfähig bleiben. Falls doch die Notwendigkeit besteht, über größere Entfernungen zu kommunizieren, sollten die Verbindungen dynamisch zur Laufzeit gewählt werden und nicht vorher festliegen.
- **Skalierbarkeit:** Die Lösung sollte in der Lage sein, Hunderte oder gar Tausende von Knoten zu integrieren und zu verwalten.
- **Kompatibilität:** Ein nahtloser Übergang zu drahtgebundenen Netzwerken sollte möglich sein, so dass so genannte "*wired-cum-wireless environments*" aufgebaut werden können. Für den Fall, dass beispielsweise in Gebäuden doch Infrastruktur zur Verfügung steht, sollte das System in der Lage sein, diese auszunutzen, um Skalierbarkeit und Durchsatz zu erhöhen und den Ressourcenverbrauch der mobilen Knoten zu verringern.
- **Effizienz:** Die Übertragungsbandbreite in drahtlosen Netzwerken ist in der Regel geringer als in drahtgebundenen und die Wahrscheinlichkeit von Interferenzen steigt mit der Knotendichte. Die entwickelte Lösung sollte effizient sein, insbesondere bezüglich der Nutzung der Übertragungsbandbreite.

Ein Protokoll für die Dienst-Entdeckung kann nicht völlig unabhängig vom zugrunde liegenden Routing-Protokoll betrachtet werden, besonders wenn man Optimierungen über die verschiedenen Protokollschichten ausnutzen möchte. Hier wird ein positionsbasiertes Routing-Protokoll verwendet (siehe 2.1), da Roboter ohnehin ihre Position innerhalb ihrer Umgebung und relativ zueinander kennen müssen, um navigieren und kooperieren zu können. Ein wesentlicher Nachteil von positionsbasierten Routing-Algorithmen ist normalerweise die Notwendigkeit, vor dem Verbindungsaufbau die geografische Position des Zielknotens in Erfahrung bringen zu müssen. Dies stellt in diesem Fall jedoch keinen Nachteil dar, da ohnehin ein geeigneter Knoten lokalisiert werden muss, der den gewünschten Dienst zur Verfügung stellen kann. Die geografische Position des Dienstleisters kann als Teil der Antwort an den Kunden mitgeschickt werden. Für die Lösung wird angenommen, dass alle Knoten ihre Position innerhalb eines gemeinsamen Koordinatensystems kennen und sich in etwa in einer Ebene befinden, der dreidimensionale Fall wird hier nicht betrachtet. Außerhalb von Gebäuden könnte diese Information durch ein satellitenbasiertes System wie das Global Position System (GPS) bereitgestellt werden. Für die Positionsbestimmung in Gebäuden wurden in der Literatur verschiedene Systeme vorgestellt, beispielsweise basierend auf RFID-Chips [BM04] oder Landkarten in Kombination mit Ultraschallsensoren [LR02]. Eine kurze Übersicht geeigneter Systeme zur Positionsbestimmung für Roboter findet sich in 4.1.2.

Weiterhin wird vorausgesetzt, dass die Knoten über synchronisierte Uhren verfügen. Die Anforderungen an die Synchronität ist dabei nicht besonders hoch, der Fehler kann im Sekundenbereich (<=1s) liegen. Diese Forderung ist nötig, um Dienste und Hauptknoten mit einer Lebensdauer versehen zu können. Bei Nutzung von GPS steht ohnehin ein sehr genaues globales Zeitsignal zur Verfügung (siehe 4.1.2). In Innenräumen können entweder möglicherweise vorhandene fest installierte Zeitgeber oder Algorithmen für die Zeitsynchronisierung in (drahtlosen) Computernetzwerken [EGE02, CW04] verwendet werden.

4.1.1 Kapazität von Ad-hoc-Netzwerken

In diesem Abschnitt wird kurz darauf eingegangen, wie die Übertragungskapazität von Ad-hoc-Netzwerken gesteigert werden kann und welche Anforderungen sich daraus an den Entwurf von Netzwerkprotokollen ergeben.

In [GK00], [LBCL+01] und [GGK01] werden durch theoretische Herleitungen, Simulationen bzw. reale Experimente gezeigt, dass die Übertragungskapazität in funkbasierten Ad-hoc-Netzwerken sehr niedrig sein kann und nur schlecht skaliert. Es bezeichne n die Anzahl der Knoten im Netzwerk und W die Datenrate der Kommunikationsmodule der Knoten. In [GK00] wurde durch theoretische Herleitung gezeigt, dass bei zufälliger Auswahl der Kommunikationspartner im Netzwerk die Übertragungskapazität, die jedem Knoten für die Übertragung eigener Daten zur Verfügung steht, etwa

$$CCN_t \approx W/\sqrt{n} \tag{4.1}$$

betragen kann. Dies wird im optimalen Fall erreicht, wenn die Platzierung der Knoten und die Wahl der Sendereichweiten sowie der Sendezeiten von außen bestimmt werden können und die Mobilität der Knoten sowie der Aufwand für Routing-Algorithmen nicht berücksichtigt werden. Als Interferenzmodell wird dabei das so genannte Protokoll-Modell verwendet, nach dem die Übertragung eines Senders korrekt empfangen wird, wenn kein anderer Sender geografisch näher am Empfänger ist. Bei realen Experimenten [GGK01] skalierte die Übertragungskapazität pro Knoten mit etwa

$$CCN_r \approx W/n^{1,68} \tag{4.2}$$

Die Hauptursachen für die schon bei kleinen Knotenzahlen deutlich unter der Datenrate liegenden Übertragungskapazität sind einerseits die Nutzung eines gemeinsamen Übertragungsmediums durch alle Knoten und der daraus folgenden Interferenz, wenn mehrere Sender versuchen gleichzeitig zu senden. Andererseits besteht für Knoten in Multi-Hop-Netzwerken die Notwendigkeit, Nachrichten aneinander weiterzuleiten, so dass die Datenrate eines Knotens nicht ihm alleine zur Verfügung steht, sondern auch dazu genutzt werden muss, um Nachrichten anderer Knoten weiterzuvermitteln.

Ansätze, um die Skalierbarkeit von mobilen Ad-hoc-Netzwerken zu erhöhen, gibt es auf der physikalischen Ebene durch gerichtete Antennen, auf der Routing-Ebene durch dynamische Anpassung der Sendeleistung, oder – relevant für den Entwurf von Protokollen zur Dienst-Entdeckung – auf der Anwendungsebene durch weitgehende Beschränkung des Nachrichtenverkehrs auf lokale Kommunikationsverbindungen.
Durch gerichtete Antennen wird eine bessere Ausnutzung des beschränkten Übertragungsmediums erreicht und die Kapazität des Netzwerks kann deutlich gesteigert werden [Shep96]. Eine weitere Möglichkeit die Übertragungskapazität des Netzwerks zu steigern, besteht in der dynamischen Anpassung der Sendeleistung der Knoten. Das Verschicken einer Nachricht über eine Distanz d in nur einem Sprung verursacht bei Verwendung omnidirektionaler Antennen Kosten proportional zu d^2, da andere Nachrichtenübertragungen im gleichen Frequenzband in einem Gebiet proportional zu d^2 während dieser Zeit gestört werden. Stattdessen kann für die Weiterleitung jedoch auch beispielsweise immer der geografisch nächste Nachbar in Richtung des Ziels ausgewählt und die Sendeleistung so angepasst werden, dass der für die Weiterleitung ausgewählte Knoten die Nachricht gerade noch empfängt. Dieser Ansatz kann problemlos mit einem positionsbasierten Routing-Algorithmus wie GPSR kombiniert werden. Es wird also nicht derjenige Knoten für die Weiterleitung gewählt, der die Distanz zum Zielpunkt minimiert (wie ursprünglich bei GPSR vorgesehen), sondern derjenige Knoten, der den kleinstmöglichen Schritt in Richtung des Zielpunkts ermöglicht. Durch die erhöhte Anzahl benötigter Sprünge bis zum Zielknoten wird die Latenz erhöht, aber die Interferenz zwischen den Knoten kann gesenkt und damit die Gesamtkapazität des Netzwerks erhöht werden [Shep96]. In [LBCL+01] wird schließlich gezeigt, dass die Kapazität pro Knoten in Funknetzwerken bei einem Wachsen des Netzwerks vom Kommunikationsmuster abhängt. Die Skalierbarkeit wird von der Lokalität des Nachrichtenaufkommens bestimmt. Damit ein Ad-hoc-Netzwerk skalierbar ist, muss die mittlere Entfernung zwischen Quelle und Senke klein bleiben, der Nachrichtenaustausch also möglichst lokal beschränkt sein.

4.1.2 Systeme zur Positionsbestimmung für Roboter

Die Bestimmung der eigenen Position ist eines der grundlegenden Probleme in der Robotik und wurde bereits in einer Vielzahl von Arbeiten behandelt. Für die Anwendung von CSD wird davon ausgegangen, dass ein geeignetes

Positionsbestimmungssystem zur Verfügung steht. Ohne ein solches System könnten die Roboter ohnehin nicht in ihrer Umwelt navigieren und mit anderen Robotern kooperieren. Hier werden exemplarisch einige Systeme zur Positionsbestimmung vorgestellt, die für CSD geeignet wären.
Für Außenbereiche ist in den meisten Fällen ein satellitengestütztes System wie das US-amerikanische GPS eine mögliche Wahl [USCG]. GPS basiert auf der Messung der Signallaufzeiten von speziellen GPS-Satelliten zum Empfänger, aus denen dann die Position des Empfängers auf der Erde bestimmt werden kann. Ein weiterer Vorteil bei Verwendung des GPS-Systems ist die Verfügbarkeit einer globalen Zeit. GPS-Empfänger empfangen die Satellitenzeit und die Differenz der Satellitenzeit zur Koordinierten Weltzeit und können so die Koordinierte Weltzeit (UTC) mit Genauigkeiten von deutlich unter 1 Mikrosekunde berechnen [HC02].

Für Innenbereiche oder Gebiete in denen GPS nicht zur Verfügung steht, sind verschiedene andere Ansätze denkbar. [BM04] untersucht die Positionsbestimmung von mobilen Robotern basierend auf einer großflächigen Verteilung von RFID-Chips (Radio Frequency Identification). RFID-Chips sind miniaturisierte Transponder, die eine (kleine) Menge an Daten speichern und durch geeignete Sende-Empfangs-Einheiten ausgelesen oder beschrieben werden können. Derartige RFID-Chips können beispielsweise in Teppiche eingewebt werden und auf Anfrage durch ein RFID-Lesegerät ihre Position innerhalb des Gebäudes zurückliefern.
Ein weiteres Beispiel ist die Verwendung von Ultraschall-Sensoren in Kombination mit vorgefertigten Karten [LR02]. In diesem Projekt wurde ein autonomer Rollstuhl verwendet und in einem großflächigen Gebiet lokalisiert. Insgesamt wurde eine Wegstrecke von 2,18 km in 75 Minuten zurücklegt, bei einer maximalen Geschwindigkeit von 0,84 m pro Sekunde. In den meisten Situationen blieb der Lokalisierungsfehler deutlich unter 5 m. Dieser Ansatz ist unter anderem deshalb interessant, weil er in einer kombinierten Umgebung aus Außen- und Innenbereich getestet wurde.

Systeme zur Positionsbestimmung weisen oft Messfehler auf, die relativ zur Größe von mobilen Robotern (Größenordnung ca. 1m) nicht mehr vernachlässigbar sind. Die Auswirkungen von Fehlern in der Positionsbestimmung auf CSD werden in 4.2.3.1 eingehender diskutiert. Grundsätzlich lässt sich sagen, dass ein

Positionsbestimmungssystem für CSD ausreichend genau ist, wenn der mittlere Fehler des Positionsbestimmungssystems relativ zur Sendereichweite der Roboter klein bleibt.

4.2 Das Cell-based Service Discovery-Protokoll

A. Zellstruktur und Hauptknoten

Ein aus Zellen bestehendes Gitter bildet die Grundstruktur der Lösung. Die Positionsdaten der Roboter werden dazu verwendet, um sie geografischen Zellen zuzuordnen. Jede Zelle hat einen Hauptknoten (wenn sich Knoten in der Zelle befinden). Bei der Konstruktion des Gitters und bei niedrigen Knotendichten übernimmt der erste aktive Knoten in einer Zelle diese Rolle. Die feste Gitterstruktur hat den Vorteil, dass die Auswahl eines Hauptknotens schneller und mit weniger Nachrichtenaufwand erfolgen kann. Weiterhin kann die Auswahl je nach lokalem Bedarf erfolgen und wirkt sich nicht auf das gesamte Netz aus. Es besteht die Möglichkeit, zwischen den Zellen eine Übergangszone zu definieren, so dass Knoten, die sich in der Nähe von Zellgrenzen aufhalten oder sich entlang von Zellgrenzen bewegen, nicht ständig ihre Zellzugehörigkeit ändern. Ein Knoten ist solange Teil einer Zelle, solange er den Randbereich nicht verlassen hat. Wenn ein gewöhnlicher Knoten seine Zelle verlässt, meldet er sich von seinem Hauptknoten ab. Informationen über die Knoten einer Zelle und den von ihnen angebotenen Dienste sind mit der Zelle selbst assoziiert und nicht mit dem aktuellen Hauptknoten der Zelle. Wenn ein Hauptknoten seine Zelle verlässt, dann verbleibt die Information mit der Mehrheit der Knoten in der Zelle und wandert nicht mit ihm. Dies erfordert jedoch einen Übergabemechanismus, wenn ein Hauptknoten seine Zelle oder das Netzwerk verlässt, oder der Hauptknoten entscheidet, dass er seine Rolle nicht mehr wahrnehmen kann, zum Beispiel auf Grund niedriger Energiereserven (Abb. 4-1, untere Zelle). Anstatt einen kompletten Wahlzyklus innerhalb der Zelle auszuführen, wählt der Hauptknoten einen Nachfolger aus, basierend auf ihm zur Verfügung stehenden Daten wie den Positionen der Knoten in seiner Zelle oder ihrer bisherigen Aufenthaltsdauer in der Zelle. So wird in der Simulation (Kapitel 5) zum Beispiel ein Knoten in der Nähe der geografischen Zellmitte als Nachfolger bevorzugt, weil er leichter von den anderen Knoten kontaktiert werden kann und länger braucht, um die Zelle zu verlassen. Eine Übergabe-Nachricht benennt

den neuen Hauptknoten, optional auch eine Liste der Knoten in der Zelle einschließlich ihrer Positionen und angebotenen Dienste sowie die aktuell laufenden Suchprozesse. Für das positionsbasierte Routing tauschen benachbarte Knoten regelmäßig Funkfeuer-Nachrichten aus, die ihre Position enthalten. Diese Funkfeuer werden auch dazu verwendet, um Informationen über die aktuelle Position und Identität des Hauptknotens innerhalb der Zelle zu verbreiten. Wenn der Ausfall eines Hauptknotens durch das Ausbleiben regelmäßiger Aktualisierungen festgestellt wird, wird der erste Knoten, der sich zum Nachfolger benennt, der neue Hauptknoten. Im Falle eines Konflikts kann eine einfache Entscheidungsstrategie angewendet werden, beispielsweise basierend auf der eindeutigen Identifikationsnummer der Knoten. Die Verwendung von Hauptknoten kann, muss jedoch kein Nachteil sein. Bei der durchaus realistischen Annahme einer Heterogenität der Knoten kann diese beispielsweise ausgenutzt werden, indem leistungsfähigere Knoten als Hauptknoten ausgewählt werden. Insbesondere wird die Integration von dedizierten drahtgebundenen Knoten erleichtert, die als immobile permanent verfügbare Hauptknoten dienen können. Dies ermöglicht eine einfache Ausnutzung vorhandener Infrastruktur.

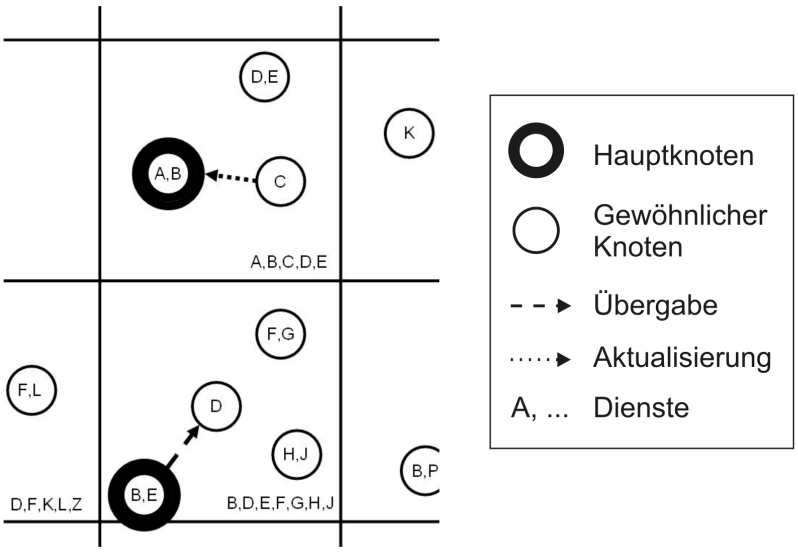

Abbildung 4-1. Illustration des Nachrichtenaustauschs in Zellen. In der oberen Zelle sendet der Knoten, der Dienst C anbietet, eine Aktualisierung seiner Position an seinen Hauptknoten. In der unteren Zelle verlässt der Hauptknoten seine Zelle und wählt den Knoten, der Dienst D anbietet, als seinen Nachfolger. Die Dienste, die in einer Zelle verfügbar sind, werden in der unteren rechten Ecke der Zelle zusammenfassend aufgeführt.

B. Dienstsuchen

Nun wird der Suchprozess betrachtet. Ein neuer Knoten taucht im Netzwerk auf und möchte einen Dienst nutzen, beispielsweise ein Roboter, der gerade aktiviert wurde. Unter Verwendung seiner Positionsdaten kann er sich einer geografischen Zelle zuordnen. Wenn es in seiner Zelle bereits einen Hauptknoten gibt, wird der Knoten kurz nach seiner Aktivierung dessen Position und Identität über Funkfeuer-Nachrichten von seinen Nachbarn erhalten. Wenn es keinen Hauptknoten in der Zelle gibt, ernennt sich der neue Knoten selbst dazu. Im Folgenden betrachten wir den umfassenderen Fall, dass es bereits einen Hauptknoten M in der Zelle gibt. M kennt alle verfügbaren Dienste in seiner Zelle. Abhängig von der Zellgröße können nicht alle Knoten durch einen einzigen Nachrichtensprung mit ihren Hauptknoten kommunizieren. Daher müssen die Knoten regelmäßig, beispielsweise zeit- und/oder ereignisgetriggert, ihren Hauptknoten kontaktieren und ihre aktuelle Position übermitteln. Zusammen mit ihrer Position schicken sie eine Auflistung ihrer aktuell angebotenen Dienste (Abb. 4-1, obere Zelle). Der suchende Knoten schickt eine Anfrage an M, der zuerst seine Tabelle der verfügbaren Dienste prüft, um die Anfrage direkt zu beantworten. Falls M den gesuchten Dienst nicht in seiner eigenen Zelle findet, leitet er die Anfragen an benachbarte Zellen bis zu einer bestimmten geografischen Distanz weiter (Abb. 4-2).

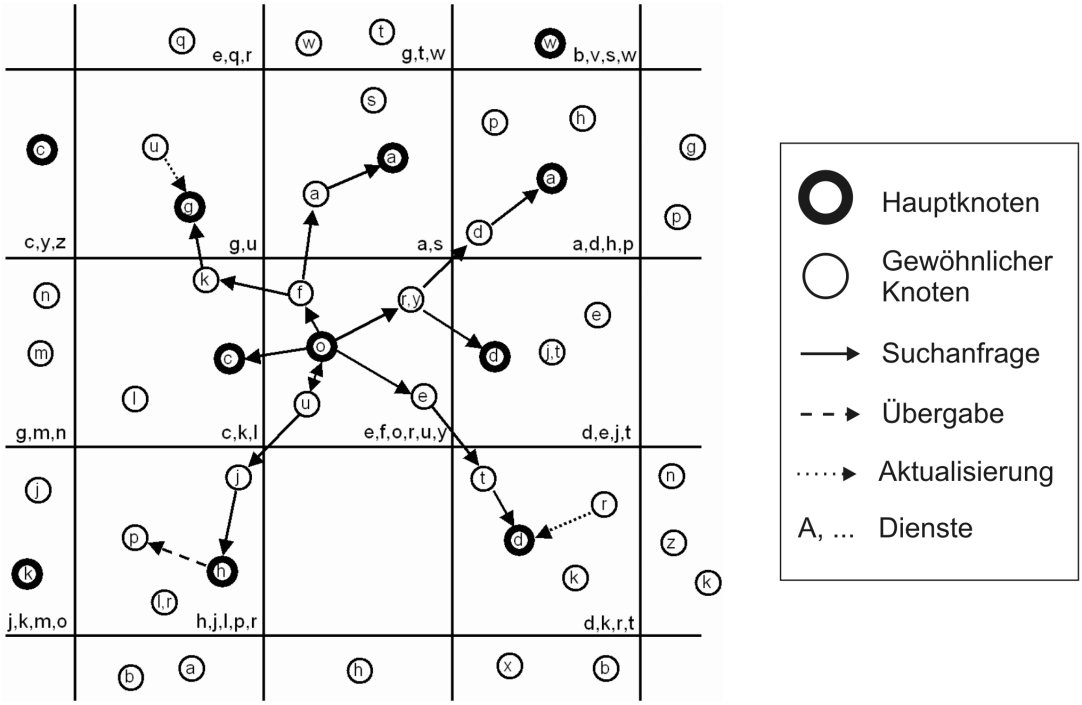

Abbildung 4-2. Illustration der Suche zwischen Zellen. Der untere linke Knoten der mittleren Zelle sendet eine Suchanfrage an seinen Hauptknoten. Dieser sendet Anfragen an seine acht direkten Nachbarzellen.

Je nach Latenzanforderung können optional mehrere Suchanfragen gebündelt werden. Interzell-Kommunikation basiert auf Zell-IDs, weil die Hauptknoten wechseln können. Da alle Knoten ein gemeinsames Koordinatensystem teilen, genügt es eine Zell-ID anzugeben, um eine Nachricht durch einfache geografische Weiterleitung an die Zielzelle zu senden. Bei Eintreffen der Nachricht werden die Knoten der Zelle die Nachricht an ihren Hauptknoten weiterleiten, da sie seine Position kennen. Abhängig von der erwarteten Anzahl der Repliken eines Dienstes im Netzwerk oder wenn ein Hauptknoten weiß, dass sich ein gesuchter Dienst in der näheren Nachbarschaft aufhält und daher einen geringen Suchradius erwartet, wird eine expandierende Suche durchgeführt. Dabei werden die ersten Suchanfragen nur an die direkten Nachbarn geschickt. Wenn die Suchanfrage bei den direkten Nachbarzellen fehlschlägt, werden die Nachbarn zweiter Ordnung befragt. Diese expandierende Suche wird solange ausgeführt, bis der gewünschte Dienst lokalisiert oder bis eine maximale Anzahl an Suchschritten überschritten ist. Weiterhin kann optional positionsbasiertes Multicasting für eine effiziente Verteilung der Suchnachrichten verwendet werden. In [MHWL03] wird ein Multicast-Protokoll vorgestellt, das positionsbasierte Unicast-Protokolle wie GPSR erweitert, indem Regeln für das Duplizieren von Nachrichten basierend auf ausschließlich lokalen Informationen integriert werden. Nachdem ein Dienst erfolgreich

lokalisiert wurde, wird die Position des Dienstleisters an M geschickt und von M an den anfragenden Knoten in seiner Zelle weitergeleitet.

C. Klassifizierung

In [MWH01] wurde eine Klassifikation eingeführt, die Lösungen für das Suchen von Knoten im Netzwerk danach unterscheidet
a) welche Menge M von Knoten Informationen über andere Knoten anbieten und
b) über wie viele Knoten des Netzwerks jeder Knoten aus M Informationen anbietet.

Solch eine Klassifikation ist auch für Lösungen zur Dienst-Entdeckung sinnvoll. Es wird zwischen vier verschiedenen Ansätzen unterschieden.

- *Some-for-some*: Einige Knoten des Netzwerks bieten Informationen über einige Knoten des Netzwerks.
- *Some-for-all*: Einige Knoten des Netzwerks bieten jeweils Informationen über alle Knoten des Netzwerks.
- *All-for-some*: Jeder Knoten des Netzwerks bietet Informationen über einige Knoten des Netzwerks.
- *All-for-all*: Jeder Knoten des Netzwerks bietet Informationen über jeden Knoten des Netzwerks.

CSD fällt dabei in die Kategorie some-for-some. Die Hauptknoten jeder Zelle haben Informationen über alle Knoten ihrer Zelle.

Eine andere mögliche Form der Klassifizierung ist die Einordnung der Lösung in die Kategorien zentral bzw. verteilt. Demnach wäre CSD ein hybrides Verfahren mit einer zentralen Organisationsstruktur innerhalb der Zellen und einer verteilten Suche außerhalb.

D. Kommunikations-, Speicher- und Zeitkomplexität

In Tabelle 4-1 ist die Kommunikations-, Speicher- und Zeitkomplexität von CSD dargestellt. Für eine einfache Vergleichbarkeit wurden die Werte für die anderen Lösungsverfahren aus 2.2.2.2 ebenfalls aufgeführt. Die Tabelle dient dazu, eine qualitative Einschätzung der verschiedenen Lösungsarten zu ermöglichen. Da hier Netzwerkgrößen von einigen hundert bis tausend Knoten betrachtet werden, spielen die in der Landau-Notation weggefallenen Konstanten für einen quantitativen Vergleich durchaus eine Rolle. Quantitativ wird CSD in 4.3 und Kapitel 5 in der Simulation untersucht.

Es sei n die Anzahl der Knoten im Netzwerk und m die mittlere Anzahl von Diensten pro Knoten. Für eine erste Analyse werden einige vereinfachende Annahmen getroffen. Interferenzen bei der Kommunikation werden nicht berücksichtigt. Es wird davon ausgegangen, dass jede versendete Nachricht auch korrekt empfangen wird. Die Auswirkungen von Interferenzen werden in Kapitel 5 in der Simulation berücksichtigt. Auch Repliken werden zu Beginn nicht betrachtet, jeder Dienst ist im Netzwerk einzigartig. Es wird davon ausgegangen, dass die Knoten in etwa gleich auf einer annähernd quadratischen Grundfläche verteilt und Kommunikationsverbindungen bidirektional sind. Die Knoten bilden ein Gitter aus $\sqrt{n} \cdot \sqrt{n}$ Knoten. Bei beschränkten Sendereichweiten der Knoten sind $O(\sqrt{n})$ Sprünge nötig, um eine Nachricht quer durch das Netzwerk zu schicken.
Weiterhin wird angenommen, dass die Knotendichte beschränkt ist und Zellen (soweit zutreffend) eine feste Größe haben, so dass die maximale Anzahl von Knoten in einer Zelle ebenfalls beschränkt ist. Bei der Betrachtung des Nachrichtenaufwands wird beim Versenden einer Nachricht über eine Multi-Hop-Verbindung jeder Sprung für sich gezählt. Hierdurch wird die tatsächliche Belastung des Netzwerks besser widergespiegelt und die Kommunikationsdistanz fließt bei der Kostenbetrachtung mit ein. Lösungsspezifische Kostenpunkte, im Fall von CSD die Wahl der Hauptknoten (bei LANES die Verwaltung der Bahnen und bei Rendezvous Regions die Wahl der Server), werden nicht berücksichtigt.

Installation pro Dienst gibt die Anzahl der Nachrichten an, die ein neu eingetroffener Dienst benötigt, um sich im Netzwerk bekannt zu machen. Bei CSD schickt jeder

Knoten Bekanntmachungen an den jeweiligen Hauptknoten seiner Zelle. Wenn die Knotendichte beschränkt ist und die Zellen eine feste Größe haben, dann existiert eine konstante obere Schranke, die die maximale Anzahl der benötigen Hops angibt, um eine Nachricht von jedem beliebigen Knoten der Zelle an den Hauptknoten zu senden. Gesamtinstallation gibt die Gesamtzahl an Nachrichten an, die benötigt werden, um ein Netzwerk aus n Knoten zu initialisieren. Dabei wird angenommen, dass zuerst alle n Knoten positioniert und aktiviert werden und sich die Knoten dann nacheinander bekannt machen. Dies sind die Minimalkosten für die Initialisierung des Netzwerks. Wenn die Knoten nacheinander eingeschaltet werden, sind die Kosten höher. Jeder Knoten bietet im Mittel m Dienste an. Es müssen also n*m Dienste bekannt gemacht werden. Dies multipliziert mit dem jeweiligen Aufwand für die Bekanntmachung eines Dienstes ergibt die Grundkosten für die Gesamtinstallation. Aktualisierung pro Dienst gibt die Anzahl der Nachrichten an, die benötigt werden, um Aktualisierungen eines Dienstes, z. B. bei Positionsänderungen, im Netzwerk zu verbreiten. Für CSD entsprechen die Kosten denen der Installation. Suche pro Dienst gibt die Anzahl der Nachrichten an, die benötigt werden, um einen Dienst im Netzwerk zu lokalisieren. Im Fall von CSD müssen im schlechtesten Fall alle Zellen befragt werden und deren Anzahl ist proportional zu n. Die Latenz gibt die Zeit an, die benötigt wird, um einen Dienst zu lokalisieren. Diese beträgt bei CSD $O(\sqrt{n})$, weil das Gebiet parallel durchsucht wird. Speicher pro Knoten gibt an, wie viele Speicherplätze für Dienste ein Knoten des Netzwerks bei der jeweiligen Lösung bereitstellen muss. Wenn die maximale Anzahl von Knoten in einer Zelle beschränkt ist, dann gibt es eine konstante obere Schranke, die abhängig von m ist und den Speicherbedarf der Hauptknoten in CSD angibt.

Tabelle 4-1. Nachrichten- und Speicherkomplexität sowie Latenz von CSD im Vergleich zu anderen Lösungen.

Nachrichtenaufwand / Latenz / Speicheraufwand	CSD	LANES	GCLP	Rendezvous Regions	GLS[1]
Installation / Dienst	O(1)	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\log(n)\sqrt{n})$
Gesamtinstallation m Dienste / Knoten[2]	O(n)	$O(n\sqrt{n})$	$O(n\sqrt{n})$	$O(nm\sqrt{n})$	$O(nm \log(n)\sqrt{n})$
Aktualisierung / Dienst	O(1)	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$
Suche / Dienst	O(n)	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\log(n)\sqrt{n})$
Latenz / Suche	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(\log(n)\sqrt{n})$
Speicher / Knoten bei m Diensten / Knoten	O(m)	$O(m\sqrt{n})$	$O(m\sqrt{n})$	O(m)	O(m log (n))

4.2.1 Details

Hier werden einige Aspekte des Protokolls wie die Wahl der Hauptknoten, die Aktualisierungsrate, mit der gewöhnliche Knoten ihre Hauptknoten regelmäßig über Positions- und Dienständerungen informieren, und das Suchmuster für die Suche zwischen Zellen im Detail betrachtet.

4.2.1.1 Wahl von Hauptknoten

Die Bestimmung einer minimalen, überdeckenden Menge M von Knoten, so dass jeder Knoten des Netzwerks maximal d Sprünge von einem Knoten aus der Menge M entfernt ist, ist NP-vollständig [APVH00]. Daher wurden in der Literatur verschiedene Heuristiken vorgeschlagen, um die Wahl von dedizierten Knoten im Netzwerk zu

[1] GLS wurde ursprünglich nicht für die Entdeckung von Diensten in mobilen Ad-hoc-Netzwerken entworfen. Es wird hier dennoch anstelle des einfachen Flutens in der Tabelle aufgeführt, weil die Komplexitätswerte für das Fluten trivial sind, GLS jedoch repräsentativ für Verfahren ist, die auf verteilten Hashtabellen basieren.

[2] Es wird davon ausgegangen, dass Bekanntmachungen mehrerer Dienste eines Knotens an einen anderen Knoten in einer Nachricht zusammengefasst werden, soweit dies möglich ist.

erleichtern. Die Einteilung des Netzwerks in geografische Zonen und die Wahl eines dedizierten Knotens in jeder Zone ist eine mögliche Lösung. Die Verwendung der in CSD verwendeten Quadrate für die Einteilung der Zellen ist eher willkürlich und wurde wegen der einfachen Darstellbarkeit und Umsetzbarkeit gewählt. Theoretisch könnten auch andere geometrische Formen gewählt werden, wie zum Beispiel das Hexagon. In [AKU04] wurden analytisch und in Simulation eine Einteilung eines mobilen Ad-hoc-Netzwerks in Quadrate und der damit verbundene Einfluss auf die Weglänge des betrachteten Routing-Verfahrens sowie auf die Anzahl der Clusterköpfe untersucht. Die Autoren vergleichen die Resultate mit der optimalen Lösung und kommen zu dem Ergebnis, dass eine Einteilung in Quadrate gute Resultate liefert.

Nachdem eine Zellstruktur festgelegt ist, muss sichergestellt werden, dass sich auch in jeder Zelle ein Hauptknoten befindet. Im Idealfall wird diese Aufgabe von Knoten zu Knoten weitergereicht und jeder Hauptknoten wählt vor Verlassen seiner Zelle einen geeigneten Nachfolger aus. Es sind verschiedene Kriterien für die Auswahl eines Nachfolgers denkbar. Ein einfacher Ansatz könnte zum Beispiel auf einer eindeutigen Identifikationsnummer der Knoten basieren. Dann würde etwa stets der Knoten mit der niedrigsten Identifikationsnummer in der Zelle die Wahl gewinnen. Eine andere Möglichkeit wäre es, den Nachfolger nur unter den direkten Nachbarknoten auszuwählen, um eine Übertragung des potentiell großen Datensatzes über mehrere Sprünge zu vermeiden. Dies könnte jedoch die Wahl eines Nachfolgers am Zellrand bewirken, der schon bald die Aufgabe selbst wieder abgeben muss. Es erscheint daher sinnvoller gleich einen Nachfolger zu wählen, der diese Aufgabe möglichst lange erfüllen kann. So besteht die Möglichkeit, die Position und Geschwindigkeit der potentiellen Nachfolger bei der Auswahl zu berücksichtigen. Diese Informationen stehen dem bisherigen Hauptknoten ohnehin zur Verfügung (die Geschwindigkeit abgeleitet aus den Positionsänderungen). Dann könnte die Eignung eines Knotens als Nachfolger beispielsweise unter Verwendung folgender hier nur schematisch dargestellter Gleichung bestimmt werden

$$\text{Eignung}(t) = \frac{a}{b + u(t)} + \frac{c}{d + v(t)} \tag{4.3}$$

Dabei gibt u(t) die Distanz des Knotens zur Zellmitte zu einem Zeitpunkt t an, v(t) die Geschwindigkeit des Knotens zum Zeitpunkt t, während a, b, c und d geeignet zu wählende Konstanten darstellen. Bei Verwendung eines solchen Kriteriums könnten im Fall von CSD Knoten bevorzugt werden, die nahe der Zellmitte sind und sich nur mit geringer Geschwindigkeit bewegen. Bei einem derartig gewählten Nachfolger erhöht sich die Wahrscheinlichkeit, dass dieser lange Hauptknoten bleibt. In der Simulation (Kapitel 5) bestimmen Hauptknoten ihre Nachfolger basierend auf ihrer Entfernung zur Zellmitte.

Bei einem unvorhergesehenen Ausfall eines Hauptknotens oder bei der Initialisierung des Netzwerks kann es allerdings passieren, dass eine Zelle keinen amtierenden Hauptknoten hat. Dann müssen die gewöhnlichen Knoten der Zelle selbstständig einen neuen Hauptknoten bestimmen. In diesem Fall kann allerdings kein Kriterium wie in Gl. (4.3) für die Eignung als Hauptknoten verwendet werden. Das Kriterium sollte sich während des Wahlvorgangs nicht ändern. In diesem Fall ist stattdessen ein einfacher Ansatz basierend auf der Identifikationsnummer der Knoten sinnvoll. Der neu gewählte Hauptknoten selbst kann bei der Wahl seines Nachfolgers selbstverständlich wieder die in Gl. (4.3) verwendeten Kriterien berücksichtigen.
Für die Wahl eines Hauptknotens durch die gewöhnlichen Knoten einer Zelle wird ein einfaches, verteiltes Verfahren angewendet, das mit wenigen Nachrichten auskommt. Die Wahl eines leitenden Knotens ist ein grundsätzliches Problem in verteilten Systemen. Verschiedene Lösungen werden in der Literatur dafür vorgeschlagen, unter anderem zum Beispiel der vollständig verteilte Echo-Algorithmus [Matt93]. Allerdings unterscheidet sich die Aufgabe hier in zwei Punkten. Einerseits soll nicht unbedingt ein bestimmter Knoten die Wahl gewinnen (zum Beispiel der Knoten mit der höchsten Identifikationsnummer), sondern einfach ein Knoten in jeder Zelle bestimmt werden, der die Aufgabe als Hauptknoten übernimmt und dieser den anderen Knoten bekannt gemacht werden. Als zweiter Punkt ist die Dynamik in mobilen Ad-hoc-Netzwerken zu berücksichtigen, insbesondere im Hinblick auf die Terminierung des Algorithmus. So werden beispielsweise beim Echo-Algorithmus Baumstrukturen aufgebaut mit den jeweiligen Knoten, die sich zur Wahl stellen, als Wurzel. Der Gewinner-Knoten empfängt auf allen von ihm ausgehenden Zweigen eine Bestätigungsnachricht, was für ihn das Zeichen ist, dass er gewählt wurde. Durch die Dynamik in mobilen Ad-hoc-Netzwerken können jedoch Knoten ausfallen oder sich aus dem Empfangsbereich ihres

Zweiges herausbewegen. Dann müsste die Wahl komplett neu durchgeführt werden, da nicht alle Bestätigungen erhalten würden. Es werden hier daher keine Bestätigungen in Form von Nachrichten verschickt. Stattdessen wird ein Knoten davon ausgehen, dass er gewählt wurde, wenn er von keinem besseren Knoten hört. Dadurch kann es allerdings bei einer temporären Spaltung des Netzwerks passieren, dass nicht alle Knoten von dem neuen Hauptknoten erfahren oder dass sogar mehrere Hauptknoten gewählt werden. Im ersten Fall werden die Knoten durch die für das positionsbasierte Routing ausgetauschten Funkfeuer vom neuen Hauptknoten erfahren, sobald das Netzwerk wieder verbunden ist. Im zweiten Fall wird eine Verschmelzung beider Netzwerke durchgeführt, bei der der unterlegene Hauptknoten seine Aufgabe abgibt.

Folgende Regeln bestimmen bei CSD, welcher Knoten einer Zelle als Hauptknoten fungiert:

1. Der aktuell gültige Hauptknoten einer Zelle fügt seine Identität, seine Position und einen Zeitstempel den Funkfeuer-Nachrichten hinzu, die er regelmäßig für das positionsbasierte Routing aussendet. Andere Knoten der Zelle verbreiten diese Informationen ebenfalls mit ihren Funkfeuer-Nachrichten und machen den Hauptknoten so in der Zelle bekannt.

2. Wenn ein Hauptknoten seine Zelle verlässt oder sich aus dem Netzwerk abmelden will, wählt er vorher einen Nachfolger aus (z. B. unter Verwendung von Gl. (4.3)) und übermittelt die gespeicherten Dienste und laufenden Suchanfragen an seinen Nachfolger.

3. Wenn ein Knoten neu im Netzwerk auftaucht und er eine bestimmte Zeit von keinem gültigen Hauptknoten hört, oder ein Knoten feststellt, dass der neueste Zeitstempel seines Hauptknotens abgelaufen ist, dann ruft er sich nach einer Wartezeit selbst als neuen Hauptknoten aus. Hierfür fügt er seinen Funkfeuer-Nachrichten seine eigene Identität und Position sowie einen Zeitstempel bei. Die Wartezeit kann ein Zufallswert sein oder von einem einfachen Kriterium wie dem Abstand des Knotens zur Zellmitte abhängen. Dadurch wird verhindert, dass alle Knoten der Zelle sich gleichzeitig als neuen Hauptknoten ausrufen.

4. Wenn ein Knoten neu im Netzwerk aufgetaucht ist oder ein Knoten, der einen Ausfall seines bisherigen Hauptknotens festgestellt hat, von einem (neuen) Hauptknoten mit noch gültigem Zeitstempel hört, dann stellt er sich selbst nicht mehr zur Wahl. Er akzeptiert den (neuen) Hauptknoten und verbreitet ihn sofort weiter.

5. Wenn es bei der Wahl eines neuen Hauptknotens zu einer Konkurrenzsituation zwischen zwei oder mehr Knoten kommt, wird die Identifikationsnummer der konkurrierenden Knoten verglichen. Nur der siegende Hauptknoten (z. B. der Knoten mit der kleineren Identifikationsnummer) wird weiter verbreitet. Die unterlegenen Wettbewerber akzeptieren den Sieger ebenfalls, sobald sie von ihm erfahren.

6. Wenn zwei vorher unverbundene Teilgebiete einer Zelle mit jeweils eigenen Hauptknoten zusammenkommen, kommt es ebenfalls zu einer Konkurrenzsituation. Sie wird wie in Punkt 5 behoben.

Nun wird überprüft, ob alle möglichen Szenarien abgedeckt sind.

i) Es gibt keinen Hauptknoten in der Zelle. Hier greift Regel 3. Nach einer gewissen Zeit werden sich ein oder mehr Knoten für eine Wahl zum Hauptknoten zur Verfügung stellen.
ii) Es gibt genau einen Hauptknoten in der Zelle. Hier gilt Regel 1. Dies stellt den Normalfall dar, es gibt genau einen Hauptknoten, der allen anderen Knoten der Zelle bekannt ist und diese regelmäßig bezüglich seiner Funktionsfähigkeit informiert.
iii) Es gibt mehr als einen Hauptknoten in der Zelle. Es besteht eine Konkurrenzsituation zwischen Hauptknoten. Hier greift Regel 5. Es wird eine Entscheidung basierend auf den Identifikationsnummern der Hauptknoten getroffen.
iv) Es gibt keinen Hauptknoten und keinen Kandidaten in der Zelle. Regel 3 löst diese Situation auf. Nach einer gewissen Zeit werden sich ein oder mehr Knoten für eine Wahl zum Hauptknoten zur Verfügung stellen.

v) Es gibt keinen Hauptknoten und genau einen Kandidaten in der Zelle. Hier greifen Regeln 3 und 4. Dieser Kandidat wird die Wahl gewinnen und neuer Hauptknoten der Zelle.

vi) Es gibt keinen Hauptknoten und mehr als einen Kandidaten in der Zelle. Es besteht eine Konkurrenzsituation zwischen Kandidaten bei der Wahl zum Hauptknoten. Hier greift Regel 5. Es wird eine Entscheidung basierend auf den Identifikationsnummern der Kandidaten getroffen.

vii) Der Hauptknoten verlässt geplant die Zelle. Regel 2 bewirkt, dass der Hauptknoten einen Nachfolger benennt.

viii) Der Hauptknoten verlässt wegen Ausfalls die Zelle. Hier greift Regel 3. Nach einiger Zeit werden gewöhnliche Knoten den Ausfall ihres Hauptknotens bemerken und sich selbst zur Wahl stellen.

ix) Ein Knoten taucht neu in der Zelle auf. Wenn es bereits einen oder mehr Hauptknoten in der Zelle gibt oder gerade eine Wahl durchgeführt wird, bewirken die Regeln 4 und 5, dass genau ein Knoten als Hauptknoten anerkannt wird. Wenn es noch keinen Hauptknoten gibt und auch keine Wahl durchgeführt wird, bewirkt Regel 3, dass sich der neue Knoten nach einer gewissen Zeit selbst zur Wahl stellt.

x) Ein gewöhnlicher Knoten verlässt geplant die Zelle. Dies hat keine Auswirkungen, es sei denn, dieser Knoten ist Kandidat bei einem Wahlvorgang zum Hauptknoten. Hätte er die Wahl ohnehin verloren, hat sein Weggang ebenfalls keine Auswirkungen. Hat dieser Knoten die Wahl bereits gewonnen, bewirkt Regel 2, dass er einen Nachfolger benennt. Hätte der Knoten die Wahl gewonnen, ohne dass es jedoch zu einem Abschluss des Wahlvorgangs gekommen ist, bewirkt Regel 3, dass nach einer gewissen Zeit ein neuer Wahlvorgang in der Zelle startet.

xi) Ein gewöhnlicher Knoten verlässt wegen Ausfalls die Zelle. Dies hat keine Auswirkungen, es sei denn, dieser Knoten war Kandidat bei einem Wahlvorgang zum Hauptknoten. Hätte er die Wahl ohnehin verloren, hat sein Ausfall keine Auswirkungen. Hat dieser Knoten die Wahl bereits gewonnen oder hätte er die Wahl gewonnen, wenn es zu einem Abschluss des Wahlvorgangs gekommen wäre, folgt aus Regel 3, dass nach einer gewissen Zeit ein neuer Wahlvorgang beginnt.

4.2.1.2 Aktualisierungsrate

In CSD aktualisieren Knoten regelmäßig ihren Hauptknoten bezüglich ihrer Position und den von ihnen angebotenen Diensten. Eine Fragestellung ist nun, wie oft solche Aktualisierungen verschickt werden sollten. Bei einer zu häufigen Aktualisierung wird das Netzwerk unnötig belastet. Andererseits könnten bei einer zu geringen Aktualisierungsrate geeignete Dienste nicht gefunden werden.
In [KMP99] werden verschiedene Aktualisierungsstrategien für mobile Ad-hoc-Netzwerke verglichen. Die Autoren untersuchen zeitbasierte, distanzbasierte und verbindungsbasierte Ansätze. Bei zeitbasierten Verfahren werden in regelmäßigen zeitlichen Abständen Aktualisierungen versendet. Bei distanzbasierten Verfahren sendet ein Knoten eine Aktualisierung aus, wenn eine bestimmte Zeit vergangen ist und er sich von dem Ort fortbewegt hat, an dem er seine letzte Aktualisierung versendet hatte. Beim verbindungsbasierten Ansatz sendet ein Knoten eine Aktualisierung aus, wenn sich eine bestimmte Anzahl von Verbindungen zu seinem Nachbarn verändert haben, sei es durch das Hinzukommen neuer Verbindungen oder den Abbruch alter Verbindungen. Bei diesem Ansatz unterscheiden die Autoren noch zwischen Verfahren basierend auf der absoluten Zahl (eine Aktualisierung erfolgt, wenn eine bestimmte, feste Anzahl von Verbindungsänderungen geschehen sind) und der relativen Zahl von Verbindungen (eine Aktualisierung erfolgt, wenn ein bestimmter Prozentsatz der Verbindungen sich geändert hat). Die Autoren halten als Ergebnis ihrer Untersuchungen fest, dass für unstrukturierte mobile Ad-hoc-Netzwerke der verbindungsbasierte Ansatz mit absoluter Zählung der Verbindungsänderungen die besten Ergebnisse erzielt. Sie weisen jedoch ebenfalls darauf hin, dass für zellulär organisierte Netzwerke der distanzbasierte Ansatz besser geeignet ist.
Aktualisierungsstrategien für zelluläre Netzwerke wurden in [BKS94] analytisch untersucht. Für ihre Analyse verwenden die Autoren ringförmig angeordnete Zellen, nehmen jedoch an, dass ihre Ergebnisse auch für gitterförmig angeordnete Zellen aussagekräftig sind. Sie vergleichen unter anderem distanzbasierte und zeitbasierte Verfahren und kommen zu dem Schluss, dass distanzbasierte Verfahren in den von ihnen betrachteten Szenarien die besten Ergebnisse erzielen.

Für CSD ist ein modifizierter distanzbasierter Ansatz sinnvoll. Ein Knoten sendet grundsätzlich in bestimmten Zeitabständen Aktualisierungen aus, auch wenn er sich

nicht bewegt hat. Dies ist notwendig, um unvorhergesehene Ausfälle von Knoten durch andere Knoten detektieren zu können. Falls sich Knoten aber durch das Netzwerk bewegen, wird gegebenenfalls auch schon früher eine Aktualisierung verschickt, wenn eine bestimmte Distanz zum Ort der letzten Aktualisierung zurückgelegt wurde. Dies ist gerade bei Einsatz von positionsbasiertem Routing und bei Austausch von positionsabhängigen Diensten wie in der Robotik wichtig.

4.2.1.3 Suchmuster und expandierende Suche

Ein Knoten, der einen Dienst sucht, befragt als erstes den Hauptknoten seiner Zelle. Wenn der Hauptknoten keinen geeigneten Dienst kennt, so wird er eine Suchanfrage an andere Zellen des Netzwerks schicken. Es sind unterschiedliche Strategien denkbar, um auszuwählen, an welche Zellen des Netzwerks die Suchanfrage geschickt werden soll. Grundsätzlich sollte im hier betrachteten Szenario versucht werden, die Suchanfragen möglichst lokal zu halten. Dadurch werden einerseits die im Vergleich zu infrastrukturbasierten Netzwerken hohen Kosten für Kommunikation über große Entfernungen gering gehalten, andererseits ist dann die geografische Distanz zwischen Kunden und Dienstleistern bei physisch zu erbringenden Diensten nicht so groß. Daher erscheint es sinnvoll, die Suchanfrage in konzentrischen Ringen um die suchende Zelle zu verbreiten.

In [CH05] wurde analytisch und in Simulation untersucht, ob das Durchsuchen eines Netzwerks in mehreren Schritten in Form einer expandierenden Suche oder das Durchsuchen in einem Schritt effizienter ist. Die Autoren untersuchen sowohl große wie auch kleine Netzwerke und unterscheiden dabei Netzwerke mit und ohne Cachespeichern in den Knoten. Als Ergebnis halten die Autoren fest, dass das Durchsuchen des Netzwerks ohne Caches in mehreren Suchschritten unter den von ihnen betrachteten - relativ allgemeinen - Randbedingungen in den meisten Fällen nur wenig Ersparnisse bringt, aber die Latenz deutlich erhöht. Das Bild ändert sich, wenn Caches bzw. Repliken eingeführt werden. Dann kann durch eine expandierende Suche eine signifikante Ersparnis in der Anzahl der Nachrichten erzielt werden.

Bei Anwendung einer expandierenden Suche sind zwei Möglichkeiten denkbar, um Suchnachrichten nach einem erfolglosen ersten Suchschritt zu verbreiten. Entweder

wird das bereits durchsuchte Gebiet nochmals abgesucht, das heißt, es findet einfach eine Wiederholung des gleichen Suchvorgangs mit einem größeren Suchradius statt. Alternativ kann beim folgenden Suchschritt das bereits durchsuchte Gebiet ausgelassen werden. Dann werden eine oder mehrere Suchnachrichten zuerst an den Rand des bereits durchsuchten Gebiets gesendet und erst dann verbreitet. Im ersten Fall sind höhere Suchkosten zu erwarten. Vorteile des ersten Verfahrens sind seine größere Zuverlässigkeit. Im zweiten Fall sind die Suchkosten geringer. Allerdings sind potentiell mehr Fehlerfälle möglich. So könnte sich ein Knoten genau dann vom noch nicht durchsuchten Gebiet in das durchsuchte Gebiet bewegen, nachdem der erste Suchschritt vollführt ist, aber der zweite noch nicht begonnen hat. In der Simulation wird dennoch die zweite Variante angewandt. Wie man später sehen wird, wirken sich die theoretischen möglichen Fehlerfälle kaum aus und können anderweitig umgangen werden.

4.2.2 Mögliche Erweiterungen

Hier werden mögliche Erweiterungen für CSD betrachtet, ein Cache-System und der aktive Austausch zwischen Zellen auch ohne Suchanfragen.

4.2.2.1 Cache-System

In diesem Abschnitt wird ein mögliches Cache-System für CSD beschrieben. Gerade in funkbasierten mobilen Ad-hoc-Netzwerken, in denen einerseits das Versenden von Nachrichten teuer sein kann, auf der anderen Seite aber Nachbarn Nachrichten mithören können und Nachrichten aneinander weiterleiten, erscheint die Verwendung von Caches in den Knoten sinnvoll. In den Cache-Speichern werden Informationen gespeichert, die während der Nutzung von CSD weitgehend ohne zusätzlichen Nachrichtenverkehr erhalten werden können, wie beispielsweise durch Auswertung weitergeleiteter Nachrichtenpakete.

Die Möglichkeit, Caches bei Dienst-Entdeckungen zu nutzen, hängt auch davon ab, ob Suchantworten einfach nur aus einer kurzen Bestätigung unter Angabe der Identität des gefundenen Dienstleisters und seiner Position bestehen oder ob genaue

Dienstbeschreibungen als Suchantworten zurückgegeben werden. Beide Fälle sind denkbar. Ein Kunde wird genau abwägen, ob ein gefundener potentieller Dienstleister tatsächlich geeignet ist und ob dieser in Anspruch genommen wird. Im ersten Fall muss der Kunde den Dienstleister kontaktieren, um nähere Informationen zu erhalten, im zweiten Fall kann er basierend auf der erhaltenen Dienstbeschreibung entscheiden und braucht den Dienstleister nur zu kontaktieren, wenn er diesen auch in Anspruch nehmen möchte. Nur im zweiten Fall sind Caches bei semantisch komplexen Dienstanfragen von Nutzen. Denn nur mit einer Dienstbeschreibung können Knoten überprüfen, ob ein Dienst zu einer empfangenen Suchanfrage passt, um diese dann gegebenenfalls aus dem Cache zu beantworten.

Bei der Entwicklung eines Caches fallen einige grundsätzliche Entwurfsentscheidungen an, die die Speicherstruktur, den Lesevorgang, den Schreibvorgang und die Löschung von Cache-Einträgen betreffen [BB03]. Die Speicherstruktur beschreibt, welche Informationen gespeichert und in welcher Form sie im Cache vorgehalten werden. So besteht beispielsweise im Fall von CSD die Möglichkeit, zu einem gegebenen Dienst den anbietenden Knoten mitsamt Position zu speichern, oder aber auch nur die Zelle, in der der Dienst angeboten wird.
Die Regeln für das Schreiben in den Cache bestimmen, wann ein Eintrag in den Cache aufgenommen wird. So könnte ein Knoten in CSD beispielsweise nur diejenigen Dienste in seinen Cache schreiben, die er auch selbst genutzt hat. Zusätzlich könnte er auch die durch ihn weitergeleiteten Nachrichtenpakete auswerten und so seinen Cache aktualisieren. Schließlich bestünde für den Knoten als umfassendste Option auch die Möglichkeit, den um ihn stattfindenden, nicht an ihn adressierten Nachrichtenverkehr mitzuhören, und damit seinen Cache zu füllen. Die Regeln für das Auslesen des Caches bestimmen, wann auf einen Cache-Eintrag zurückgegriffen werden darf. Ein Knoten wird in den meisten Fällen seinen eigenen Cache benutzen, bevor er beginnt, einen Dienst zu suchen. Eine andere Frage ist, ob ein Knoten Suchanfragen anderer Knoten aus seinem Cache beantworten oder sogar eine über ihn geleitete Suchantwort verändern darf, wenn er meint, aktuellere Informationen über den Dienstleister in seinem Cache zu haben. Als letzte grundsätzliche Entwurfsentscheidung bestimmen die Regeln für die Löschung von Einträgen aus dem Cache, welche Einträge aus dem Cache-Speicher entfernt werden und wann dies geschieht. Dies kann beispielsweise geschehen, wenn eine Nachricht mit der Aufforderung zum Austragen eingeht, wenn

der Cache-Speicher voll ist oder wenn eine bestimmte Haltezeit abgelaufen und der Cache-Eintrag veraltet ist.

Für CSD wird ein verteiltes, zeitbasiertes Cache-System vorgeschlagen. Die Verwendung aufwendiger, zentral oder hierarchisch koordinierter Cache-Strategien führt in mobilen Ad-hoc-Netzwerken aufgrund ihrer Verteiltheit und Dynamik zu vergleichsweise hohen Kosten, um die Caches aktuell und konsistent zu halten.
Der Cache-Speicher ist in CSD vollständig verteilt, jeder Knoten verfügt über einen eigenen Cache. Die Verwaltung von Diensten durch Hauptknoten und von Diensten im Cache-Speicher unterscheiden sich nicht, abgesehen davon, dass die Dienste im Cache-Speicher mit einer niedrigeren Gültigkeitsdauer versehen werden. Dies vereinfacht die Implementierung des Cache-Speichers. Im Cache-Speicher werden zu einem Dienst der zugehörige Knoten und dessen Position gespeichert.

Bezüglich des Schreibens in den Cache werden die von einem Knoten selbst genutzten Dienste in seinen Cache geschrieben. Zusätzlich sind zwei Möglichkeiten denkbar. Um eine Suchantwort an den Kunden zurückzuschicken, wird die Nachricht entweder einfach durch den zugrunde liegenden geografischen Routing-Algorithmus auf dem kürzesten Weg zurück zum Kunden geleitet oder aber es wird die Randbedingung eingeführt, dass bei der Durchquerung einer Zelle die Nachricht auch stets über den Hauptknoten dieser Zelle geleitet werden muss. Im ersten Fall ist der zu erwartende Weg kürzer. Im zweiten Fall werden die Caches der Hauptknoten stets auf einem aktuellen Stand gehalten, dies ist jedoch mit einer hohen Belastung und einer erhöhten Wahrscheinlichkeit für Interferenzen in der Umgebung der Hauptknoten verbunden. Die erste Variante ist in diesem Fall daher empfehlenswerter. Die Knoten, die Suchantworten weiterleiten, extrahieren und schreiben Informationen über die Dienste und deren Zellen in ihren eigenen Cache und informieren ihren Hauptknoten im Rahmen des normalen Aktualisierungsvorgangs. Sobald diese Information im Cache des Hauptknoten steht, steht sie allen Knoten der Zelle zur Verfügung, da Suchanfragen stets über ihn geleitet werden. Ein Nachteil bei diesem Verfahren ist, dass es zu einer gewissen Verzögerung kommen kann, bevor diese Cache-Information anderen Knoten des Netzwerks über den Hauptknoten zur Verfügung steht. Die Dauer dieser Verzögerung hängt von der Aktualisierungsrate der Knoten ab.

Für das Schreiben in den Cache ergibt sich insgesamt also folgendes Bild. Ein Dienst, der selbst genutzt wurde, wird in den Cache übernommen. Ebenso werden Suchantworten von den Knoten ausgewertet, die sie weiterleiten. Es wird nicht gefordert, Nachrichten stets über den Hauptknoten der Zelle zu leiten. Eine Auswertung mitgehörter, aber nicht durch den mithörenden Knoten selbst weitergeleiteter Nachrichtenpakete findet nicht statt.

Ein Knoten verwendet stets seinen eigenen Cache, bevor er einen Suchvorgang auslöst. Ebenso beantwortet ein Knoten Suchanfragen anderer Knoten aus seinem Cache.

Es werden keine expliziten Nachrichten zur Entwertung von Caches gesendet, dies wäre in einem mobilen Ad-hoc-Netzwerk nachrichtenaufwendig und könnte einen Großteil der Vorteile der Verwendung von Caches wieder zunichte machten. Stattdessen werden Bekanntmachungen von Diensten mit einer Gültigkeitsdauer versehen, und es wird eine zeitbasierte Entwertung vorgenommen. Die Verwendung eines zeitlichen Kriteriums ist ein einfacher Ansatz für die Entwertung von Cache-Einträgen, der keinen zusätzlichen Nachrichtenaufwand erzeugt. Bei vollem Cache-Speicher werden ebenfalls die ältesten Einträge überschrieben. Der zeitliche Grenzwert muss geeignet gewählt werden, da verfrühte oder verspätete Entwertungen der Cache-Einträge die Performanz des Protokolls sogar noch verschlechtern können (analytische Verfahren zur Bestimmung geeigneter Haltezeiten für Cache-Einträge werden in dieser Arbeit allerdings nicht entwickelt). Im Fall einer verfrühten Löschung wird ein Cache-Eintrag gelöscht, obwohl die darin enthaltenen Informationen eigentlich noch immer gültig wären. Im schlechtesten Fall, wenn alle Einträge stets zu früh gelöscht werden, wirkt es so, als würde das System einfach ohne Cache arbeiten. Dies hat nicht so gravierende Auswirkungen wie verspätete Löschungen. Im Fall einer verspäteten Löschung verweilt der Eintrag im Cache, obwohl die darin enthaltenen Informationen bereits veraltet und nicht mehr gültig sind. Hier können im schlechtesten Fall Anfragen regelmäßig falsch beantwortet werden, was in mehreren Kontaktversuchen und aufwendigen Korrekturen resultiert. Grundsätzlich ist es umso schwieriger, einen geeigneten Wert zu finden, je dynamischer das Netzwerk in seiner Struktur und je dynamischer der über ihn geleitete Nachrichtenverkehr ist. Es wäre denkbar, die zeitlichen Grenzen für Löschungen adaptiv anzupassen. Dies schlägt sich jedoch wiederum in einem erhöhten

Koordinationsaufwand nieder und erfordert komplexere Strategien, die jedoch ebenfalls nicht im Rahmen dieser Arbeit liegen.

4.2.2.2 Erweiterung auf Fünfergruppen von Zellen

Die Anzahl der Zellen, an die eine Suchanfrage gesendet wird, steigt schnell mit der Entfernung zur Ursprungszelle an. Durch eine zusätzliche pro-aktive Komponente ist es möglich, die Anzahl der Zellen zu senken, an die Suchanfragen gesendet werden müssen, um ein bestimmtes Gebiet zu durchsuchen. Dabei tauscht jede Zelle in regelmäßigen Zeitabständen und/oder ereignisgesteuert ihre Daten mit ihrer oberen, unteren, linken und rechten Nachbarzelle aus (Abb. 4-3). Die Wahl von vier statt beispielsweise acht Nachbarzellen ist willkürlich. Je weniger dynamisch das Netzwerk, desto größere Gebiete können in diesen Austausch zwischen Zellen einbezogen werden. Dieser Austausch kann völlig asynchron und verteilt geschehen. Damit kennt jede Zelle auch die Knoten und Dienste ihrer vier direkten Nachbarn. Dieser Ansatz ist vergleichsweise robust, da jede Zelle weiterhin auch allein voll funktionsfähig ist, unabhängig vom Zustand seiner Nachbarzellen. Es besteht bei zufälliger Gleichverteilung der Dienste nun eine fünfmal so hohe Wahrscheinlichkeit, dass bereits die Anfrage beim eigenen Hauptknoten direkt zum Erfolg führt, weil dieser nun die Dienste der Knoten aus fünf Zellen kennt. Damit verringert sich die Anzahl der Nachrichten, die für das Durchsuchen eines Gebiets erforderlich sind, ebenfalls auf ein Fünftel, weil nur noch jede fünfte Zelle befragt werden muss. Es wird also eine Zunahme von pro-aktiven, lokalen Nachrichten in Kauf genommen, um eine Abnahme der reaktiven, nicht-lokalen Nachrichten zu erreichen. Je nach Häufigkeit von Suchanfragen und dem Verhältnis von Suchanfragen zu Knotenbewegungen kann insgesamt eine Verringerung des Nachrichtenaufkommens erzielt werden.

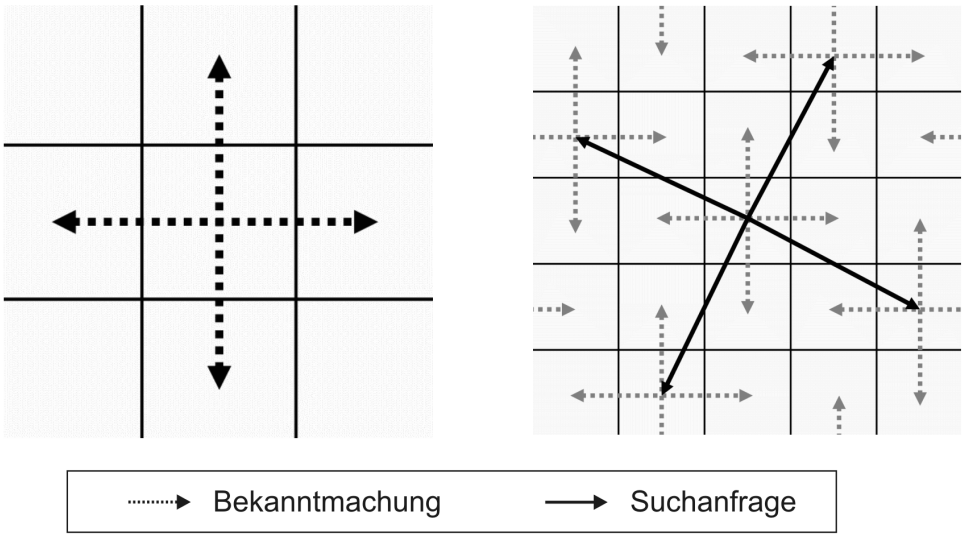

Abbildung 4-3. Jede Zelle tauscht aktiv Informationen über die in ihr enthaltenen Dienste mit ihrem oberen, unteren, linken und rechten Nachbarn aus (links). Dies verringert den Aufwand für Suchanfragen, erhöht allerdings den Aufwand für Bekanntmachungen. Im rechten Bild ist das Suchmuster für die mittlere Zelle dargestellt, sie muss für eine Suche nur noch jede fünfte Zelle befragen. Suchende Zellen schicken ihre Suchanfragen so aus, dass die entsprechenden Fünfergruppen um sie herum zentriert sind.

Wenn nun ein Dienst benötigt wird, der weder in der suchenden Zelle Z_S selbst noch in den jeweils zwei horizontalen und vertikalen Nachbarzellen zur Verfügung steht, werden Suchanfragen an benachbarte Fünfergruppen ausgesendet. Die Zelle Z_S wählt dabei die Zellen, an die sie ihre Anfragen schickt so, dass die entsprechenden Fünfergruppen um sie zentriert sind und eine vollständige Abdeckung des Suchgebiets gewährleistet ist (Abb. 4-3). Auch hier kann Z_S die Fünfergruppen im gewünschten Suchgebiet entweder in nur einem Suchschritt oder durch eine expandierende Suche mit zunehmendem Suchradius absuchen.

Die Entscheidung für eine Aktivierung dieser Zwischenschicht muss von außen getroffen werden. Bei einem Einsatz dieser Komponente muss sie für das gesamte Operationsgebiet aktiviert werden. Denn einerseits müssen sich alle benachbarten Zellen beim Austausch der Aktualisierungen beteiligen, andererseits sollte jeder Zelle bekannt sein, dass Suchanfragen nur noch gezielt an jede fünfte Zelle im Netzwerk geschickt werden müssen. Da eine Entscheidung zur Aktivierung dieser Komponente das gesamte Netzwerk betrifft und keine Zelle für sich einen Überblick über den Gesamtzustand des Netzwerks hat, muss sie von außen, beispielsweise bei der

Initialisierung des Netzwerks, getroffen werden. Je nach Verhältnis von Bekanntmachungen zu Dienstsuchen lohnt sich der Einsatz dieser Erweiterung ab 16 bis 50 Zellen. Eine Abschätzung hierfür findet sich in A.6.

Es sei angemerkt, dass durchaus auch andere Möglichkeiten denkbar sind, um zusätzliche pro-aktive Komponenten in CSD einzuführen. Die Grundversion von CSD bildet ein 2-Schichten-Modell, bestehend aus normalen Knoten in der unteren und Hauptknoten in der höheren Schicht. Nun wäre es denkbar, eine weitere Hierarchisierung beispielsweise in Form eines Quadtrees wie beim Grid Location Service (siehe 2.2.2.2) einzuführen. Dabei würden vier Zellen zu einer Zelle der nächsthöheren Ordnung vereinigt, mit jeweils einem Hauptknoten für jede Ordnung. Bei einer zunehmenden Anzahl von Schichten wird insgesamt jedoch die Robustheit der Lösung abnehmen, weil ein Knotenausfall auf einer höheren Ebene Auswirkungen auf eine Vielzahl von Knoten hätte. Weiterhin kann es zu einem erhöhten Nachrichtenaufwand für Knotenaktualisierungen kommen, da bei Knotenbewegungen die Aktualisierungen von der untersten bis zur obersten Ebene propagiert werden müssen. Schließlich werden Knoten, die für eine solche Aufgabe auf hoher Ebene ausgewählt werden, unverhältnismäßig belastet und können Flaschenhälse im Netzwerk darstellen. Eine weitere Hierarchisierung erscheint daher eher sinnvoll, wenn Infrastruktur bereitsteht. Die hier vorgestellte Lösung führt dagegen keine weitere Hierarchieebene ein, sondern arbeitet völlig verteilt. Jede Zelle bleibt für sich voll funktionsfähig, unabhängig vom Zustand seiner Nachbarzellen, und die Hauptknoten jeder Zelle bedienen auch nur die Knoten ihres Gebiets.

4.2.3 Betrachtung möglicher Fehlerszenarien

In diesem Teil werden Fehlerszenarien, wie Messungenauigkeiten bei der Positionsbestimmung der Knoten oder das Vorhandensein von Hindernissen, die die Kommunikationsverbindungen stören, und deren Auswirkungen auf CSD betrachtet.

4.2.3.1 Auswirkungen von Fehlern in der Positionsbestimmung

In diesem Abschnitt wird analysiert, wie sich Fehler der Roboter bei der Bestimmung der eigenen Position auf CSD auswirken. Dabei werden Fehler betrachtet, die im Verhältnis zur Sendereichweite der Knoten klein sind. Falls kein Funktionsfehler vorliegt, wird diese Bedingung bei Verwendung der in 4.1.2 vorgestellten Systeme zur Selbstlokalisierung erfüllt sein. Im Falle von schwerwiegenden Funktionsfehlern oder gar Ausfällen der Positionsbestimmung wird der fehlerhafte Roboter nicht mehr am Kommunikationsprozess teilnehmen und auch nicht mehr mit anderen Robotern interagieren können. Es wird davon ausgegangen, dass dieser Fall vom Roboter detektiert werden kann. Hier werden nur die Auswirkungen von Fehlern in der Positionsbestimmung auf die Funktionsfähigkeit von CSD untersucht. Die Auswirkungen von Fehlern auf das zugrunde liegende positionsbasierte Routing-Protokoll wurden im Rahmen verschiedener Lösungsansätze in der Literatur behandelt (siehe 2.1).

Zwei Fehlerfälle können in CSD unterschieden werden. Solange sich ein Fehler in der Positionsbestimmung eines Knotens K nicht so auswirkt, dass er sich einer falschen Zelle zuweist, hat er keine Auswirkungen auf CSD. An einer Zellgrenze kann es jedoch passieren, dass sich der Knoten aufgrund eines Fehlers in der Positionsbestimmung einer falschen Zelle Z zuordnet. Insgesamt sind hier jedoch ebenfalls keine weiteren Auswirkungen auf die Funktionsfähigkeit von CSD zu erwarten. K kann sich einfach regulär in der Zelle Z beteiligen (Abb. 4-4). Dies gilt sowohl für Hauptknoten wie auch gewöhnliche Knoten.

Fehler in der Positionsbestimmung haben, soweit sie klein im Verhältnis zur Sendereichweite bleiben, also keine Auswirkungen auf die Funktionsfähigkeit von CSD.

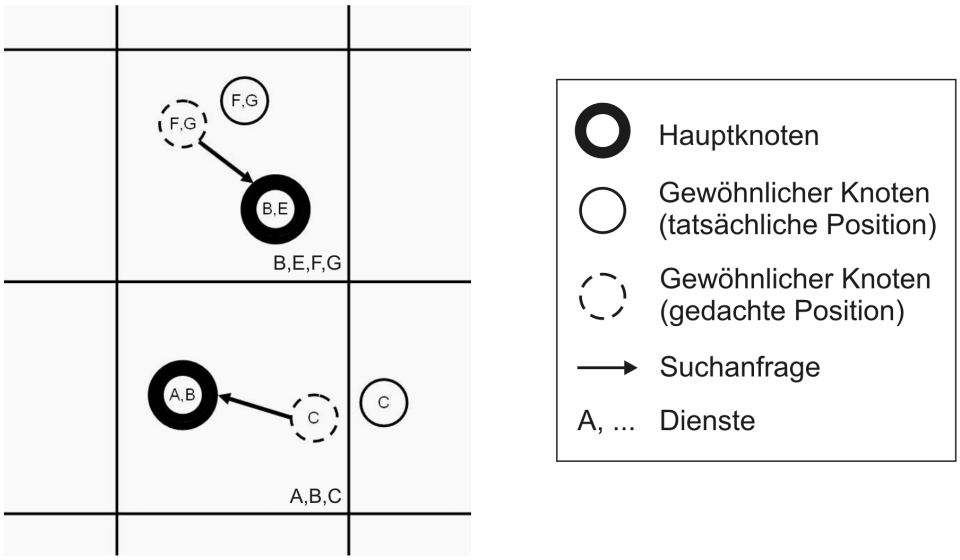

Abbildung 4-4. Fehler in der Positionsbestimmung haben keine Auswirkungen auf CSD, wenn sie klein im Verhältnis zur Sendereichweite bleiben. In der oberen Zelle ist dargestellt, wie sich ein Knoten in der eigenen Zelle falsch lokalisiert. In der unteren Zelle ist dargestellt, wie sich ein Knoten aufgrund eines Lokalisierungsfehlers einer falschen Zelle zuordnet.

4.2.3.2 Auswirkungen von Hindernissen

Ein wichtiger Punkt, der bisher in anderen Arbeiten nicht betrachtet wurde, ist die Funktionsfähigkeit der Protokolle bei Berücksichtigung von Hindernissen.

Bei LANES werden Bahnen, die durch Hindernisse geteilt werden, als separate Bahnen betrachtet. Dadurch entsteht das Problem, das Dienste jenseits eines Hindernisses nicht entdeckt werden können, auch wenn es sich um ein leicht zu umgehendes Hindernis handelt. Die in GCLP vorgeschlagene Lösung weist diesbezüglich dieselbe Eigenschaft auf. Es ist jedoch robuster, da Dienstbekanntmachungen und Suchanfragen in zwei Richtungen, vertikal und horizontal verbreitet werden. Damit steigt die Wahrscheinlichkeit, dass wenigstens einer der zwei Schnittpunkte zwischen Suchanfrage des Kunden und Dienstbekanntmachung des Dienstleisters erreichbar ist.

Bei Betrachtung der Auswirkungen von Hindernissen auf Ansätze, in denen Knoten in Zellen organisiert sind, müssen mehrere Arten von Hindernissen unterschieden werden. Große Hindernisse, die ganze Zellen ausfüllen, kleine Hindernisse, die Zellen nur teilweise ausfüllen und Hindernisse, die eine Zelle in mehrere Fragmente spalten (Abb. 4-5). Für Rendezvous Regions stellen große Hindernisse ein Problem dar, weil Ressourcen, die eigentlich auf diese Region abgebildet werden sollen, keine Knoten als

Server vorfinden. Die Autoren des Artikels schlagen als Ausweg Ersatzregionen vor [SHa04].

Für CSD stellen große Hindernisse kein Problem dar. Es gibt keine Knoten in den bedeckten Zellen und Nachrichten, die an diese Zellen gerichtet sind, werden einfach fallengelassen. Wenn ein Knoten oder eine Zelle, der bzw. die kontaktiert werden soll, jenseits eines solchen Hindernisses liegt, kann das Ziel durch den Perimeter-Modus des positionsbasierten Routing-Protokolls dennoch erreicht werden (der Perimeter-Modus von GPSR kann Nachrichten um Hindernisse herumleiten, siehe 2.1). Ein Problem stellen nur in mehrere Teilgebiete gespaltene Zellen auf, in denen sich Knoten befinden, die nicht miteinander kommunizieren können. In diesem Fall kann es passieren, dass mehrere Hauptknoten pro Zelle aktiv sind. Eine eintreffende Suchanfrage wird dann nur von einen dieser Hauptknoten empfangen. Da dieser Hauptknoten nur eine Teilmenge der in dieser Zelle verfügbaren Dienste kennt, könnte er eine negative Antwort zurückschicken, obwohl der gesuchte Dienst in der Zelle verfügbar wäre. Dieser Fall kann jedoch von den Hauptknoten derartiger Zellen detektiert werden. Ein Indikator hierfür ist, ob die Knoten, die sie verwalten, tatsächlich das gesamte Gebiet der Zelle abdecken. Da es sich hier um mobile Roboter handelt, werden die Knoten weiterhin im Allgemeinen Informationen über ihre lokale Umgebung besitzen, die sie ebenfalls nutzen können, um das Vorhandensein von Hindernissen in der Zelle festzustellen. Wenn nur ein Teilgebiet der Zelle abgedeckt wird, könnte es sich entweder um ein Hindernis handeln, das das restliche Teilgebiet vollständig bedeckt, oder um ein Hindernis, dass die Kommunikation mit dem restlichen Teilgebiet verhindert. In [Stoj04] werden drei Algorithmen vorstellt, die bei Bedarf aktiviert werden und auch in diesem Fall garantieren könnten, dass alle Hauptknoten der gespaltenen Zelle die Nachricht erhalten. Hierzu müssen die Hauptknoten, solange der Zustand der Spaltung besteht, ankommende Anfragen auch an die anderen Hauptknoten weiterleiten.

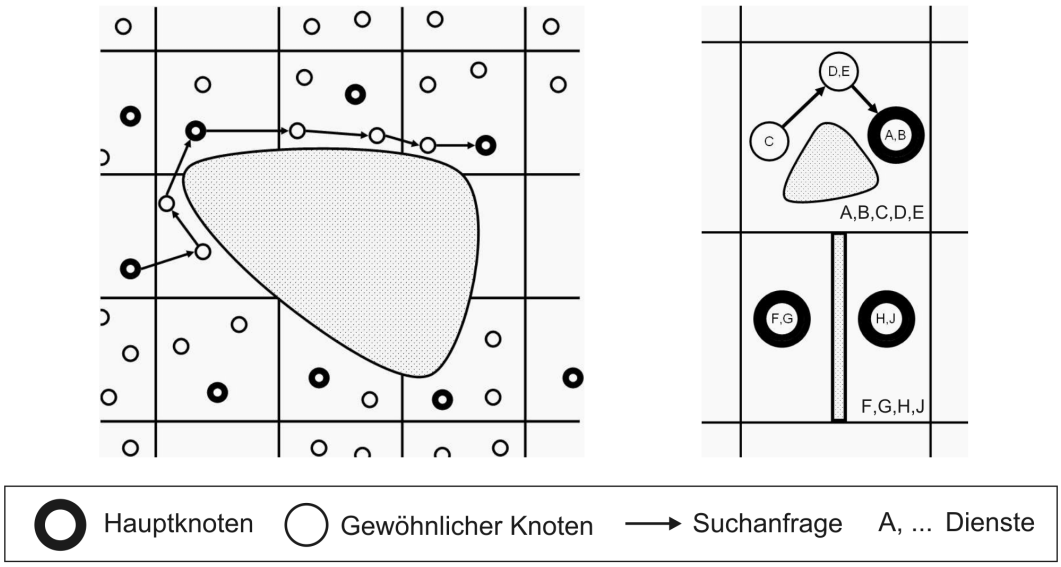

Abbildung 4-5. Hindernisse, die eine Zelle nicht spalten, haben keine Auswirkungen auf die Funktionsfähigkeit von CSD (links und rechts oben). Hindernisse, die eine Zelle spalten, können den Nachrichtenaufwand erhöhen (rechts unten). Es kann dann mehr als einen Hauptknoten pro Zelle geben, die sich untereinander austauschen müssen.

Welche Auswirkungen sind von Hindernissen zu erwarten, wenn der in 4.2.2.2 vorgestellte aktive Austausch zwischen Nachbarzellen genutzt wird? Betrachten wir wieder zuerst große Hindernisse, die eine Zelle komplett bedecken. Falls derartige Hindernisse eine oder mehrere der vier äußeren Zellen einer solchen Fünfergruppe bedecken, dann hat dies keine Auswirkungen, da diese Zellen ja ohnehin nicht abgefragt werden. Es wird wie vorgesehen der Hauptknoten der mittleren Zelle befragt. Falls jedoch die mittlere Zelle einer solchen Fünfergruppe durch ein Hindernis bedeckt ist, dann müssen die restlichen Zellen der Fünfergruppe direkt befragt werden. Dies bedeutet einfach nur eine Anwendung des Standardverfahrens, bei dem auch jede Zelle einzeln befragt wird. Hindernisse, die eine Zelle spalten, haben – abgesehen von einer kleinen Modifikation – ebenfalls keine Auswirkungen auf die Funktionsfähigkeit. Für eintreffende Nachrichten kann die oben beschriebene Lösung für die nicht auf Fünfergruppen erweiterte Version von CSD angewandt werden. Bei abgehenden Nachrichten zur Aktualisierung der Nachbarzellen kann es allerdings passieren, dass die Nachbarzellen mehrere solcher Aktualisierungen von den verschiedenen Hauptknoten der gespaltenen Zelle empfangen. Diese sollten einander nicht überschreiben. Da die Hauptknoten der gespaltenen Zelle jedoch wissen, dass sie sich in einer derartigen Zelle befinden, weil sie nur einen Teil der Knoten ihrer Zelle verwalten (siehe auch oben), können sie ihr jeweiliges Verwaltungsgebiet der Aktualisierungsnachricht beifügen.

Dann können nur Aktualisierungen von Hauptknoten mit übereinstimmendem Verwaltungsgebiet einander überschreiben.

Hindernisse wirken sich also im Prinzip nicht auf die erfolgreiche Detektion von Diensten mit CSD aus, sie können aber den Nachrichtenaufwand erhöhen.

4.3 Quantitative Betrachtung ohne Repliken

In diesem Abschnitt wird CSD detaillierter mit den anderen Lösungen verglichen. Es werden die gleichen Annahmen wie in 4.2 getroffen. Zuerst wird der Fall betrachtet, dass jeder gesuchte Dienst einzigartig ist und genau einmal im Netzwerk vorkommt.

Tabelle 4-2. Verwendete Symbole.

n	Gesamtzahl der Knoten im Netzwerk
n_c	Anzahl der Knoten in einer Zelle
b	Gesamtzahl der Dienstbekanntmachungen in CSD
k_b	Faktor zur Bewertung der Kosten von Bekanntmachungen
s	Gesamtzahl der Dienstsuchen
k_s	Faktor zur Bewertung der Kosten von Dienstsuchen
b_L	Gesamtzahl der Dienstbekanntmachungen in LANES
b_{RR}	Gesamtzahl der Dienstbekanntmachungen in Rendezvous Regions
v_{bs}	Verhältnis der Anzahl der Dienstbekanntmachungen zu Dienstsuchen in CSD
k_{bs}	Faktor für Verhältnis der Kosten von Bekanntmachungen zu Dienstsuchen
$v_{bs,L}$	Verhältnis der Anzahl der Dienstbekanntmachungen zu Dienstsuchen in LANES
$v_{bs,RR}$	Verhältnis der Anzahl der Dienstbekanntmachungen zu Dienstsuchen in Rendezvous Regions
s_{intra}	Gesamtzahl der Suchanfragen in CSD, die von Knoten an ihre Hauptknoten geschickt werden
s_{inter}	Gesamtzahl der Suchanfragen in CSD, die von Zellen an andere Zellen des Netzwerks verschickt werden
r	Anzahl der Exemplare eines Dienstes (mit gleichwertiger Funktionalität) im Netzwerk
r_e ($\leq r$)	Anzahl der Exemplare, die sich in jeweils unterschiedlichen Zellen aufhalten

4.3.1 Allgemeines

Wenn die Knoten im Netzwerk etwa gleich verteilt sind, dann ist die Anzahl der Knoten in den Zellen etwa gleich und wird mit n_c bezeichnet. Für CSD ist das Verhältnis der Anzahl der Dienstbekanntmachungen b zu der Anzahl der Dienstsuchen s (in einem gegebenen Betrachtungszeitraum)

$$v_{bs} = \frac{b}{s} \tag{4.4}$$

Entsprechend gilt für LANES

$$v_{bs,L} = \frac{b_L}{s} = \frac{b_L}{b} v_{bs} \tag{4.5}$$

und für Rendezvous Regions

$$v_{bs,RR} = \frac{b_{RR}}{s} = \frac{b_{RR}}{b} v_{bs} \tag{4.6}$$

k_b ist ein Faktor zur Bewertung der Kosten von Bekanntmachungen, k_s entsprechend zur Bewertung von Dienstsuchen. Das Verhältnis der Kosten von Dienstbekanntmachungen zu Dienstsuchen ist somit

$$k_{bs} = \frac{k_b}{k_s} \tag{4.7}$$

4.3.2 Betrachtung von CSD

Für CSD werden Suchanfragen aufgeschlüsselt nach s_{intra}, den Suchanfragen innerhalb einer Zelle, und s_{inter}, den Suchanfragen zwischen Zellen. Da jede Anfrage erst einmal an den jeweiligen Hauptknoten einer Zelle gesendet wird, gilt

$$s_{intra} = s \tag{4.8}$$

Wenn man annimmt, dass sich ein gesuchter Dienst mit gleicher Wahrscheinlichkeit über die Zellen des Netzwerks verteilt (was eher eine konservative Annahme ist, da sich Dienstleister in der Nähe ihrer Kunden aufhalten), dann gilt

$$s_{inter} = \frac{n - n_c}{n} s \tag{4.9}$$

s_{inter} spiegelt also denjenigen Anteil der Suchanfragen wieder, die nicht vom Hauptknoten der jeweiligen Zelle direkt beantwortet werden können.

Die Kosten einer Suchanfrage in CSD innerhalb einer Zelle sind proportional zu

$$K_{s,intra} = \frac{2}{3} k_s \sqrt{n_c} \tag{4.10}$$

$2/3 \cdot a$ ist die mittlere Manhattandistanz zweier zufällig gewählten Punkte auf einer quadratischen Grundfläche der Seitenlänge a (siehe A.1). Der Faktor k_s dient zur Bewertung der Kosten von Dienstsuchen. Bei n_c Knoten in jeder Zelle sind die Kosten, um eine Nachricht entlang einer Kante des Quadrats zu schicken, proportional zu $\sqrt{n_c}$. Daraus ergeben sich die in Gl. (4.10) angegebenen mittleren Kommunikationskosten. Dabei wird davon ausgegangen, dass die Knoten gitterförmig angeordnet sind, aber nicht diagonal kommunizieren können. Bei Verwendung des euklidischen Distanzmaßes sinkt die mittlere Distanz zweier Punkte auf etwa $1/2 \cdot a$ (siehe A.1). Es wird außerdem nicht berücksichtigt, dass Knoten bei einer Multi-Hop-Verbindung auch übersprungen werden können, was die Kosten senken würde.

Wenn ein Knoten in CSD seine Zelle wechselt, dann fallen zwei Aktualisierungsnachrichten an, einmal für die Abmeldung aus der alten Zelle und einmal für die Anmeldung in der neuen Zelle. Allerdings sind die Kosten für Anmeldungen, bei denen komplette Dienstbeschreibungen versendet werden, hinsichtlich der versendeten Datenmengen deutlich höher als für Abmeldungen, bei denen nur eine Kontrollnachricht ohne zusätzliche Daten übermittelt werden. Es werden hier daher (wie später bei den anderen Verfahren mit denen CSD verglichen wird auch) nur die Kosten von Anmeldungen berücksichtigt. Die Position des Knotens kann nun

nicht mehr zufällig in der Zelle gewählt werden kann, weil sich der Knoten kurz vor dem Zellwechsel an einem Zellenrand aufhalten wird. Stattdessen wird angenommen, dass sich der Knoten in einem der vier Eckenpunkte seiner Zelle befindet, während die Positionen der zwei zu benachrichtigenden Hauptknoten immer noch zufällig in ihren jeweiligen Zellen gewählt werden. Die mittlere Manhattandistanz von einer Ecke eines Quadrats der Seitenlänge a zu einem zufällig gewählten Punkt in diesem Quadrat beträgt a (siehe A.1). Mit k_b zur Bewertung der Kosten von Bekanntmachungen betragen die Kosten für einen Zellwechsel in CSD damit

$$K_{bw} = k_b \cdot \sqrt{n_c} \tag{4.11}$$

Für die Kommunikation zwischen den beiden Hauptknoten zweier Zellen gilt, dass a die mittlere Manhattandistanz zweier zufällig gewählten Punkte auf einer rechteckigen Grundfläche mit den Seitenlängen a sowie 2a ist (siehe A.1). Damit ergeben sich für die Kosten einer Suchanfrage zwischen zwei Zellen

$$K_{s,2} = k_s \cdot \sqrt{n_c} \tag{4.12}$$

Für das Durchsuchen des Netzwerks ergeben sich damit Kosten von (ohne Berücksichtigung der Kosten in der suchenden Zelle selbst)

$$K_{s,inter} = k_s \left(\frac{n}{n_c} - 1\right)\sqrt{n_c} \tag{4.13}$$

Dabei ist n/n_c die Anzahl der Zellen im Netzwerk.

Insgesamt ergibt sich als Abschätzung für die Anzahl der Nachrichten in CSD unter Berücksichtigung der Gl. (4.8) bis (4.13)

$$\begin{aligned} K_{CSD} &= K_{b,ges} + K_{s,ges} \\ &= K_{b,ges} + K_{s,intra,ges} + K_{s,inter,ges} \\ &= b \cdot K_{bw} + s_{intra}\, K_{s,intra} + s_{inter} K_{s,inter} \\ &= b \cdot k_b \cdot \sqrt{n_c} + s_{intra} \cdot \frac{2}{3} k_s \cdot \sqrt{n_c} + s_{inter} \cdot k_s \left(\frac{n}{n_c} - 1 \right) \sqrt{n_c} \\ &= b \cdot k_b \cdot \sqrt{n_c} + s \cdot \frac{2}{3} k_s \cdot \sqrt{n_c} + s \cdot k_s \left(\frac{n - n_c}{n} \right) \left(\frac{n}{n_c} - 1 \right) \sqrt{n_c} \\ &= \sqrt{n_c} \left(b \cdot k_b + \frac{2}{3} s \cdot k_s + s \cdot k_s \left(\frac{(n - n_c)^2}{n n_c} \right) \right) \end{aligned} \tag{4.14}$$

4.3.3 CSD vs. Fluten

Für das Fluten des Netzwerks betragen die Gesamtkosten

$$\begin{aligned} K_{Fluten} &= s \cdot k_s (n - 1) \\ &\approx s \cdot k_s \cdot n \qquad \text{, für } n >> 1. \end{aligned} \tag{4.15}$$

Das Verhältnis der Kosten von CSD zu Fluten ist damit

$$\begin{aligned} \frac{K_{CSD}}{K_{Fluten}} &= \frac{\sqrt{n_c} \left(b \cdot k_b + \frac{2}{3} s \cdot k_s + s \cdot k_s \left(\frac{(n - n_c)^2}{n n_c} \right) \right)}{s \cdot k_s \cdot n} \\ &= \frac{\sqrt{n_c}}{n} \left(v_{bs} k_{bs} + \frac{2}{3} + \left(\frac{(n - n_c)^2}{n n_c} \right) \right) \end{aligned} \tag{4.16}$$

4.3.4 CSD vs. LANES, GCLP und RR

Bei dieser vereinfachten Betrachtung verhalten sich LANES, GCLP und RR charakteristisch ähnlich und werden deshalb zusammen untersucht.

Bei Verwendung von LANES betragen die Gesamtkosten

$$K_{LANES} = b_L \cdot k_b \cdot \sqrt{n} + s \cdot \frac{2}{3} k_s \cdot \sqrt{n} \tag{4.17}$$

Bekanntmachungen werden im Mittel an $\sqrt{n}$ Knoten gesendet. $\sqrt{n}$ ist die mittlere Länge einer Bahn. Auch hier werden wie bei CSD nur die Kosten für Anmeldungen berücksichtigt und die Kosten für Abmeldungen aus einer Bahn vernachlässigt. Suchanfragen werden horizontal in beide Richtungen verschickt. Wenn es keine Repliken im Netzwerk gibt, dann wird die Suchanfrage in der einen Richtung bis zu derjenigen Bahn weitergeleitet, die den gesuchten Dienst enthält, und in der anderen Richtung bis zum Rand des Netzwerks. Im Mittel ergibt sich für Suchanfragen eine Kommunikationsdistanz von $2/3\sqrt{n}$ (siehe A.1).

Das Verhältnis der Kosten von CSD zu LANES beträgt damit

$$\begin{aligned} \frac{K_{CSD}}{K_{LANES}} &= \frac{\sqrt{n_c}\left(b \cdot k_b + \frac{2}{3} s \cdot k_s + s \cdot k_s \left(\frac{(n - n_c)^2}{n n_c} \right) \right)}{b_L \cdot k_b \cdot \sqrt{n} + s \cdot \frac{2}{3} k_s \cdot \sqrt{n}} \\ &= \sqrt{\frac{n_c}{n}} \frac{v_{bs} \cdot k_{bs} + \frac{2}{3} + \left(\frac{(n - n_c)^2}{n n_c} \right)}{v_{bs,L} \cdot k_{bs} + \frac{2}{3}} \end{aligned} \tag{4.18}$$

Bei GCLP sind die Kosten von Bekanntmachungen und Suchen prinzipiell höher als bei LANES, da Nachrichten sowohl vertikal als auch horizontal verschickt werden.

Die Gesamtkosten von Rendezvous Regions sind

$$K_{RR} = b_{RR} \cdot \frac{2}{3} k_b \cdot \sqrt{n} + s \cdot \frac{2}{3} k_s \cdot \sqrt{n} + b_{RR} \cdot k_b \cdot n_c \tag{4.19}$$

Analog zum Abstand zweier Knoten in einer Zelle ist die mittlere Kommunikationsdistanz zweier zufällig gewählter Knoten im Netzwerk bei angenommener quadratischer Grundfläche etwa $2/3\sqrt{n}$. Sowohl Anfragen als auch

Suchen benötigen diese Kommunikationsweite. Wie bei CSD und LANES werden die Kosten für Abmeldungen vernachlässigt. Weiterhin wird ein Term $b_{RR}*k_b*n_c$ für das Fluten der Bekanntmachung in der Zelle hinzugefügt.
Das Verhältnis der Kosten von CSD zu RR ist damit

$$\frac{K_{CSD}}{K_{RR}} = \frac{\sqrt{n_c}\left(b\cdot k_b + \frac{2}{3}s\cdot k_s + s\cdot k_s\left(\frac{(n-n_c)^2}{nn_c}\right)\right)}{b_{RR}\cdot\frac{2}{3}k_b\cdot\sqrt{n} + s\cdot\frac{2}{3}k_s\cdot\sqrt{n} + b_{RR}\cdot k_b\cdot n_c} \tag{4.20}$$

$$= \frac{\sqrt{n_c}\left(v_{bs}\cdot k_{bs} + \frac{2}{3} + \left(\frac{(n-n_c)^2}{nn_c}\right)\right)}{v_{bs,RR}\cdot\frac{2}{3}k_{bs}\cdot\sqrt{n} + \frac{2}{3}\sqrt{n} + v_{bs,RR}\cdot k_{bs}\cdot n_c}$$

4.3.5 Auswertung

Hier werden die Kostenverhältnisse von CSD zu Fluten und von CSD zu LANES betrachtet. LANES wurde ausgewählt, weil das Protokoll in dieser Betrachtung charakteristisch ähnlich zu GCLP und RR ist und bei den verwendeten Parametern innerhalb dieser Gruppe das effizienteste Verfahren und damit den stärksten Herausforderer von CSD darstellt.

Das Verhältnis von Dienstbekanntmachungen zu Dienstsuchen, v_{bs}, in dem die Mobilität und die Anfragerate der Knoten zum Ausdruck kommen, hat wesentliche Auswirkungen auf die Effizienz der einzelnen Verfahren. Da zum Beispiel CSD sehr niedrige Kosten für Dienstbekanntmachungen aufweist, aber vergleichsweise hohe Kosten für Dienstsuchen hat, ist ein hoher Wert von v_{bs} vorteilhaft für CSD. Welche Argumente sprechen in Roboternetzwerken für hohe Werte von v_{bs} und welche Argumente sprechen für niedrige Werte?

Knoten müssen regelmäßig Bekanntmachungen versenden, wenn sie sich bewegen. Für hohe Werte von v_{bs} spricht daher, dass

- Roboter, die einen Dienst erbringen, sich oft dafür auch bewegen werden. Das bedeutet, dass die meisten erfolgreichen Dienstsuchen als Folgekosten auch Bekanntmachungen nach sich ziehen werden.
- auch ohne Suchanfragen Roboterbewegungen und damit Bekanntmachungen stattfinden werden, beispielsweise durch Roboter, die Aufgaben für sich selbst erfüllen.

Für niedrige Werte von v_{bs} spricht, dass

- Knoten Dienste suchen könnten, ohne diese jemals in Anspruch zu nehmen. Beispielsweise wenn ein gewünschter Dienst nicht verfügbar ist und nach Alternativen gesucht werden muss.

Tatsächliche Werte sind stark szenarioabhängig. Deshalb werden eine Reihe verschiedener Werte für das Verhältnis von Dienstbekanntmachungen zu Dienstsuchen betrachtet.

In den Abbildungen 4-6 bis 4-11 sind die relativen Kosten von CSD zu Fluten und von CSD zu LANES für verschiedene Werte von v_{bs} (von 1:3 über 1:1 bis 3:1) und n_c (von 9 über 16 und 25 bis 36), der Anzahl der Knoten pro Zelle, dargestellt. Im Gegensatz zu CSD nutzt LANES keine Zellstruktur, um die Auswirkungen von Knotenbewegungen zu verringern, daher wurde ein Verhältnis von $b/b_L = 2$ gewählt. Der Aufwand für Bekanntmachungen der Dienste eines Knotens zum Aufwand einer Dienstsuche wurde mit $k_{bs} = 10$ berücksichtigt. Es wird also angenommen, dass das Verschicken einer Bekanntmachung, in der Dienstbeschreibungen aller Dienste eines Knotens enthalten sind, den zehnfachen Aufwand im Vergleich zum Verschicken einer Suche verursacht, in der nur die Beschreibung des gesuchten Dienstes enthalten ist.

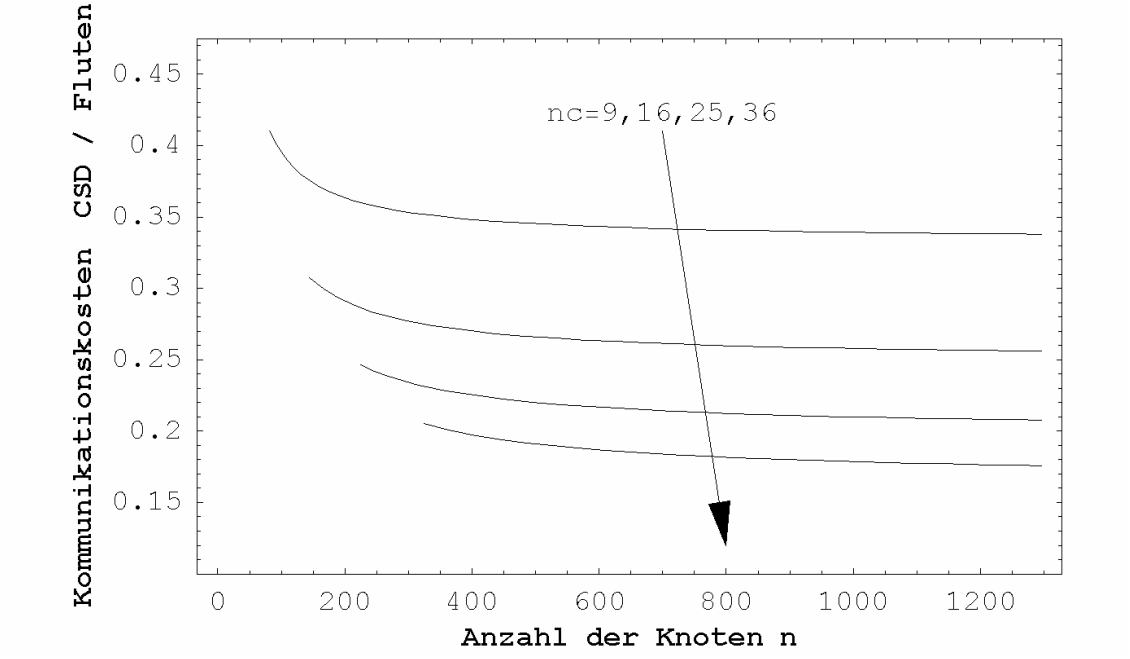

Abbildung 4-6. Vergleich der Kommunikationskosten von CSD zu Fluten (Gl. (4.16)) für verschiedene Anzahlen von Knoten pro Zelle (n_c) und einem Verhältnis von Bekanntmachungen zu Suchen (v_{bs}) von 1:3.

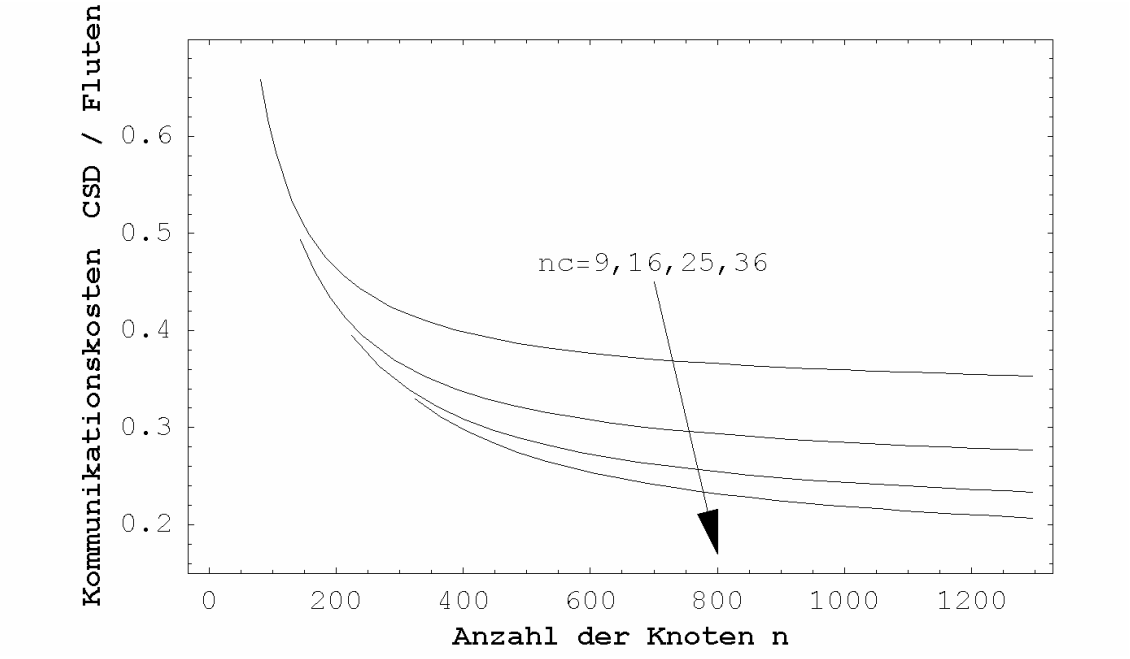

Abbildung 4-7. Vergleich der Kommunikationskosten von CSD zu Fluten (Gl. (4.16)) für verschiedene Anzahlen von Knoten pro Zelle (n_c) und einem Verhältnis von Bekanntmachungen zu Suchen (v_{bs}) von 1:1.

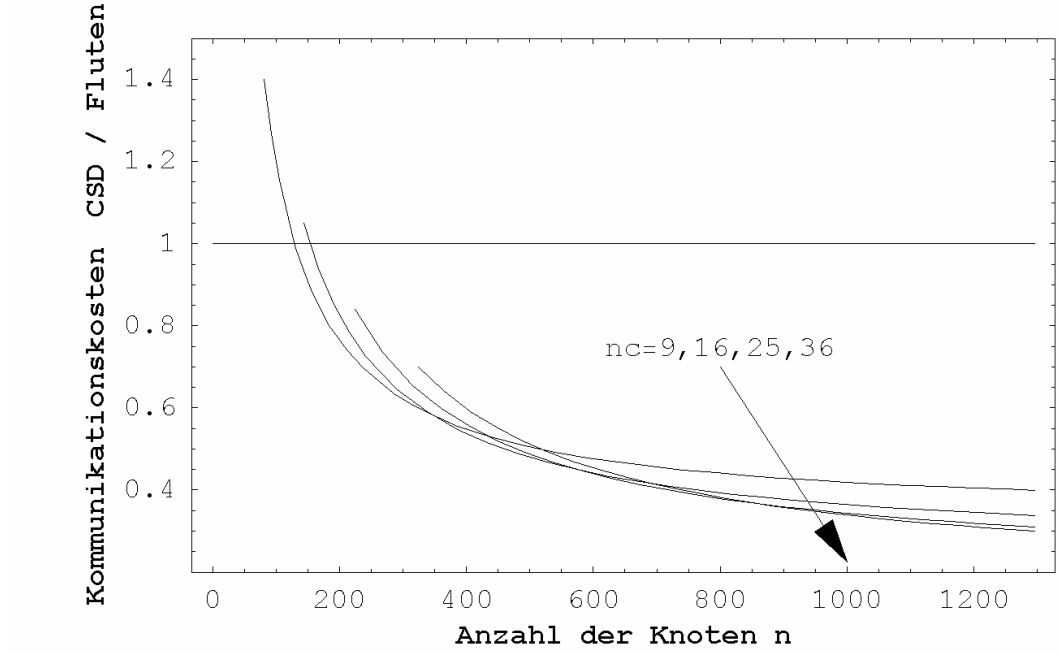

Abbildung 4-8. Vergleich der Kommunikationskosten von CSD zu Fluten (Gl. (4.16)) für verschiedene Anzahlen von Knoten pro Zelle (n_c) und einem Verhältnis von Bekanntmachungen zu Suchen (v_{bs}) von 3:1.

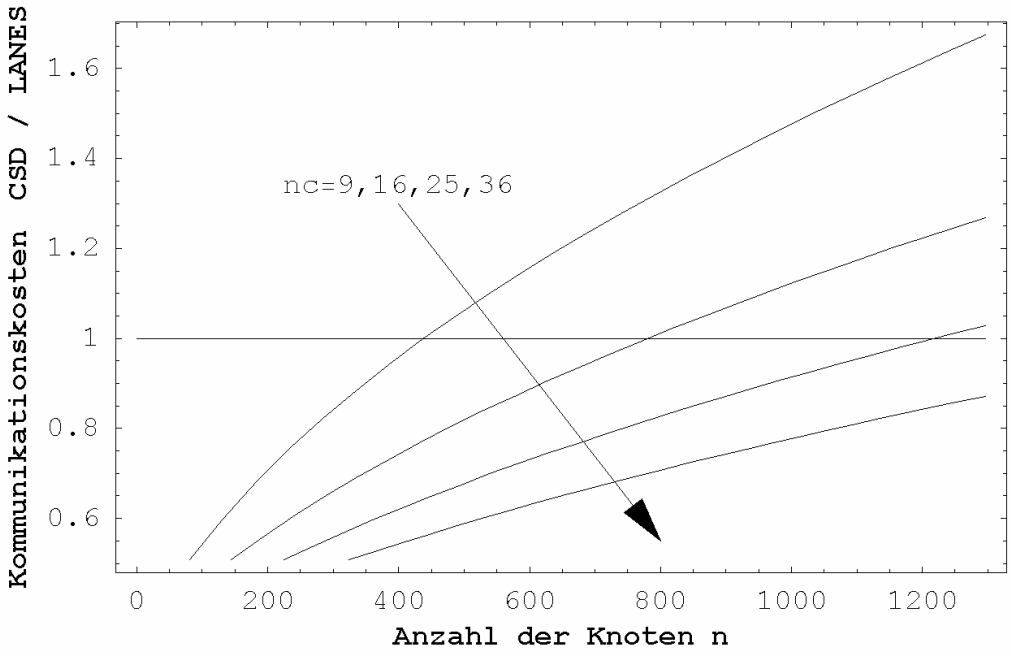

Abbildung 4-9. Vergleich der Kommunikationskosten von CSD zu LANES (Gl. (4.18)) für verschiedene Anzahlen von Knoten pro Zelle (n_c) und einem Verhältnis von Bekanntmachungen zu Suchen (v_{bs}) von 1:3

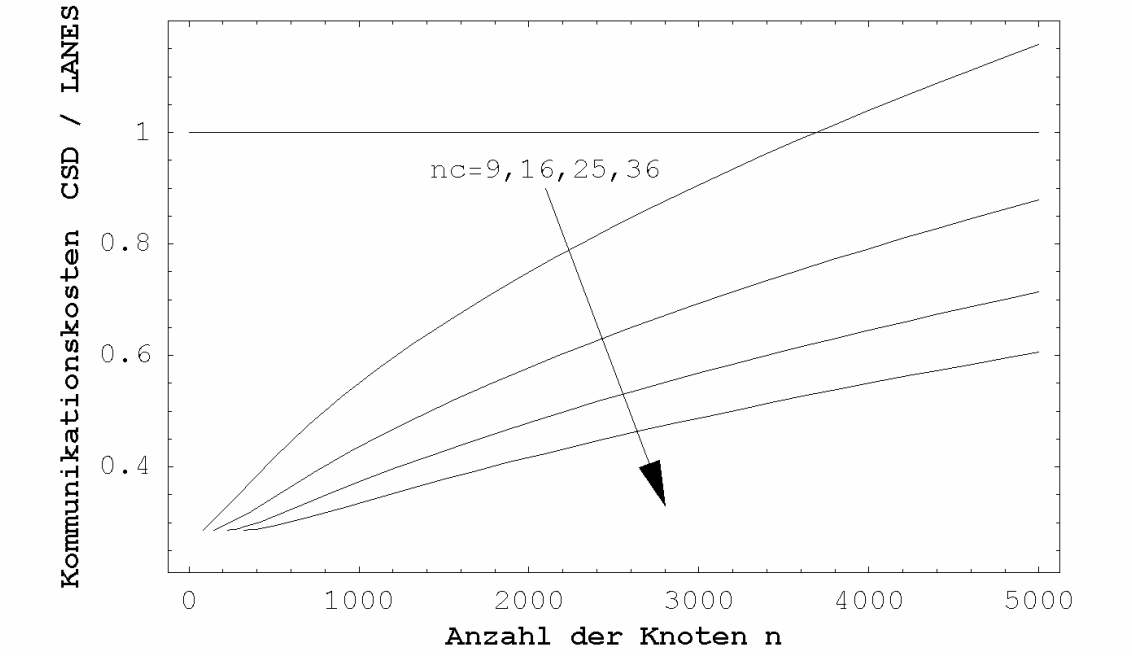

Abbildung 4-10. Vergleich der Kommunikationskosten von CSD zu LANES (Gl. (4.18)) für verschiedene Anzahlen von Knoten pro Zelle (n_c) und einem Verhältnis von Bekanntmachungen zu Suchen (v_{bs}) von 1:1

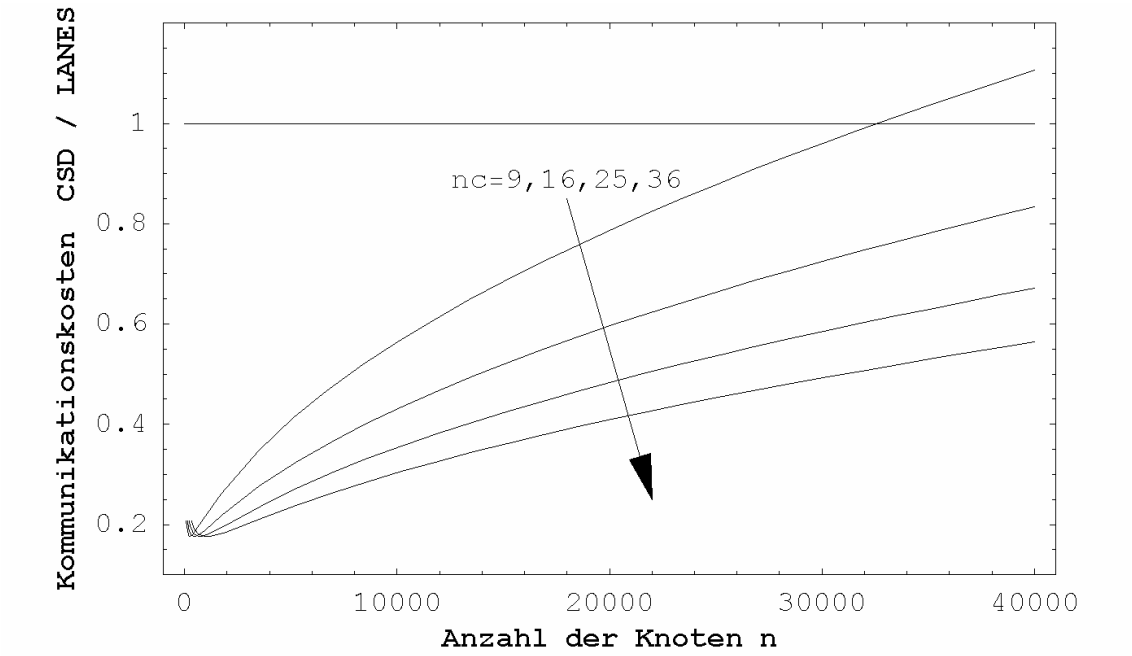

Abbildung 4-11. Vergleich der Kommunikationskosten von CSD zu LANES (Gl. (4.18)) für verschiedene Anzahlen von Knoten pro Zelle (n_c) und einem Verhältnis von Bekanntmachungen zu Suchen (v_{bs}) von 3:1

In den Abbildungen 4-6 bis 4-11 sieht man, dass CSD in den meisten betrachteten Szenarien niedrigere Kommunikationskosten als Fluten und LANES aufweist.

Ausnahmen sind sehr kleine Netzwerke mit einer niedrigen Anzahl von Dienstsuchen im Fall von Fluten und große Netzwerke mit einem niedrigen Verhältnis von Bekanntmachungen zu Suchen im Fall von LANES. Für das Fluten sind kleine Netzwerke und eine geringe Anzahl von Dienstsuchen von Vorteil, weil Suchen hohe Kosten verursachen und diese sehr schnell mit der Anzahl der Knoten steigen. Für niedrige Verhältnisse von Bekanntmachungen zu Dienstsuchen insbesondere in großen Netzwerken eignet sich LANES, da bei dieser Lösung Bekanntmachungen sehr teuer, aber Suchen vergleichsweise günstig sind. Beide Verfahren sind für ihre jeweiligen Szenarien optimiert. Für alle anderen Szenarien ist CSD geeignet und senkt die Kommunikationskosten im Vergleich zu Fluten und LANES, dies gilt ebenso für kleine Netzwerke mit vielen Dienstsuchen, mittelgroße Netzwerke im Allgemeinen wie auch für große Netzwerke mit hoher Mobilität und damit einer hohen Anzahl von Bekanntmachungen der Knoten.
Weiterhin lässt sich sehen, dass erwartungsgemäß die Kosten von CSD mit einer zunehmenden Anzahl der Knoten je Zelle (n_c) sinken. Die Messkurven für höhere Werte von n_c beginnen in den Diagrammen erst bei höheren Knotenzahlen, wenn genügend Knoten im Netzwerk sind, um wenigstens 9 Zellen zu füllen. Bei steigender Knotenzahl ist tendenziell eine Zunahme der Knoten pro Zelle und damit ein Übergang von den höheren zu den niedrigeren Messkurven bei CSD zu erwarten.

4.4 Betrachtung von CSD mit Repliken

In diesem Abschnitt werden nun Repliken, also die mögliche Existenz mehrerer gleichwertiger Exemplare eines Dienstes, berücksichtigt und CSD mit expandierender Suche betrachtet.
In einem realen Multi-Roboter-System wird die Anzahl der Dienste, die eine bestimme Funktionalität anbieten, schwanken. Es wird einzigartige Dienste geben, angeboten beispielsweise durch einen Roboter, der mit speziellen, teuren Messinstrumenten ausgestattet ist, aber auch Dienste, die nahezu jeder Roboter anbieten kann, wie beispielsweise Abstandssensoren. Grundsätzlich bewirkt die Existenz mehrerer gleichwertiger Dienste, sofern sie sich einigermaßen gleichmäßig über das Operationsgebiet verteilen, dass sich das erforderliche Suchgebiet verkleinert.

Bezüglich des Kommunikationsmusters kann im stationären Zustand von vielen nahen Anfragen und vereinzelten fernen Suchanfragen ausgegangen werden. Je öfter ein Dienst angefordert wird, desto höher wird die Wahrscheinlichkeit sein, dass dieser Dienst auch mehrmals im Netzwerk vorhanden und in der Nähe verfügbar ist. Wäre die erste Folgerung nicht erfüllt, wäre das Gesamtsystem langfristig nicht im Gleichgewicht. Wenn für bestimmte Aufgaben benötigte Dienste nicht in ausreichender Zahl im Netzwerk existieren, stellen diese Dienste einen Flaschenhals für das Gesamtsystem dar. Aufgaben, die von diesen Diensten abhängen, können dann nicht bearbeitet werden. Die zweite Folgerung ergibt sich aus der Erwartung, dass sich Dienste mit höherer Wahrscheinlichkeit in denjenigen Gebieten aufhalten, in denen sie öfter genutzt und angefordert werden. Schon aus energetischen Gründen ist anzunehmen, das Dienste bis zu ihrer erneuten Anforderung einfach in dem Gebiet verbleiben werden, in dem sie zuletzt genutzt wurden. Selbst wenn Dienste eines bestimmten Typs sich also anfangs am gleichen Ort im Netzwerk befinden, werden sie sich im Laufe der Zeit (in Abhängigkeit vom Anforderungsmuster) auf diejenigen Gebiete verteilen, in denen sie benötigt werden.

Im Folgenden werden die gleichen Annahmen wie in 4.3 getroffen und die Notation aus Tabelle 4-2 verwendet.

4.4.1 Verteilung der Repliken in den Zellen

Wenn man annimmt, dass die Repliken eines Dienstes gleich im Netzwerk verteilt sind, dann stellt sich im Fall von CSD die Frage, wie groß die Wahrscheinlichkeit ist, dass sich zwei oder mehr Repliken in der selben Zelle wiederfinden, was nur wenig Vorteile bringen würde, oder sich die Repliken alle in unterschiedlichen Zellen aufhalten. Wenn r Dienste auf n/n_c Zellen zufällig verteilt werden, wie viele Zellen enthalten im Mittel dann mindestens einen dieser Dienste?

Die Wahrscheinlichkeit, dass eine bestimmte Zelle c einen bestimmten Dienst **nicht** enthält, ist

$$P(\text{Zelle c hat einen bestimmten Dienst nicht}) = 1 - \frac{1}{n / n_c} \qquad (4.21)$$

Die Wahrscheinlichkeit, dass diese Zelle c keinen der r Dienste enthält, ist

$$P(\text{Zelle c hat keinen dieser r Dienste}) = \left(1 - \frac{n_c}{n}\right)^r \qquad (4.22)$$

Jeder dieser r Dienste verfehlt also diese Zelle c. Damit ist aber wiederum die Wahrscheinlichkeit, dass mindestens einer dieser Dienste in c ist

$$P(\text{Zelle c hat einen oder mehrere dieser r Dienste}) = 1 - \left(1 - \frac{n_c}{n}\right)^r \qquad (4.23)$$

Diese Wahrscheinlichkeit gilt für jede der n/n_c Zellen. Bei n/n_c Zellen gilt damit, dass im Mittel etwa

$$r_e = \frac{n}{n_c}\left(1 - \left(1 - \frac{n_c}{n}\right)^r\right) \qquad (4.24)$$

Zellen „gefüllt“ sind, also mindestens einen oder mehrere dieser r Dienste haben. In Abbildung 4-12 ist r_e für n = 400, n_c = 25 und r = 0..16 dargestellt. Für $r << n/n_c$ steigt die Anzahl der „gefüllten“ Zellen fast linear mit r an, die zusätzliche Repliken leisten also einen hohen Beitrag zur Reduzierung des Suchaufwands.

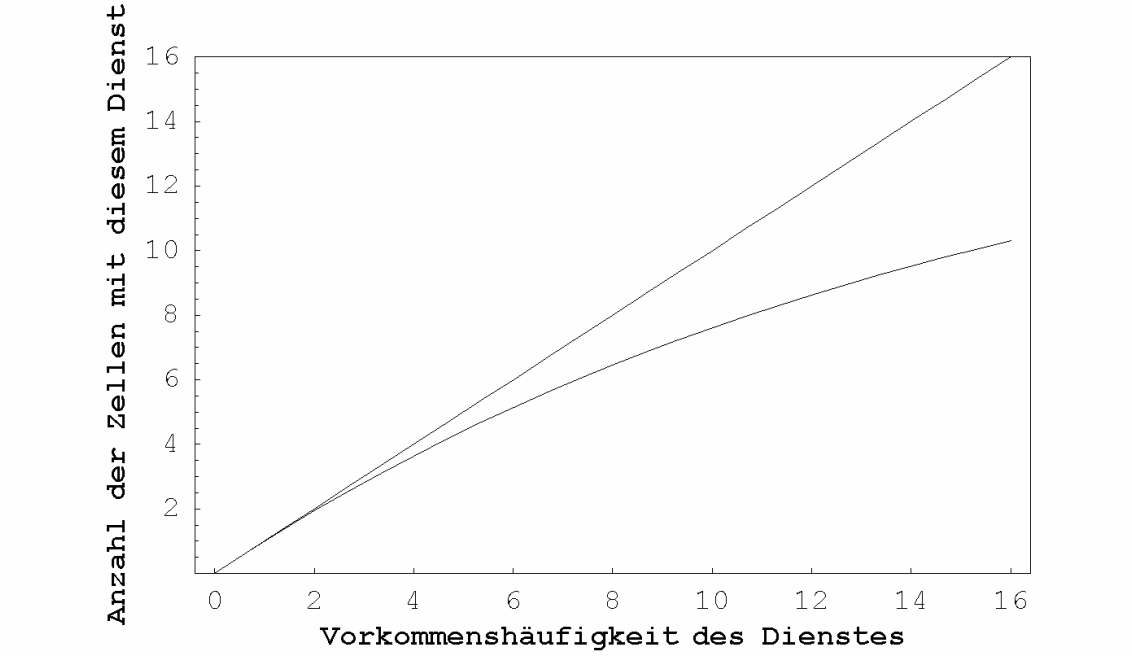

Abbildung 4-12. Wie sich mehrere Exemplare eines Dienstes auf die Zellen verteilen (untere Kurve). So ist bei bis zu vier Exemplaren die Wahrscheinlichkeit sehr hoch, dass sich diese auf vier verschiedene Zellen verteilen.

4.4.2 Expandierende Suche

Im obigen Abschnitt wurde untersucht, wie sich Repliken auf die Zellen verteilen. Mit diesem Ergebnis wird nun betrachtet, wie sich in einem Netzwerk mit Repliken durch eine expandierende Suche der Suchaufwand von CSD verringern kann. Die Betrachtung hier beschränkt sich auf zwei Suchschritte. Wie man sehen wird, können schon zwei Suchschritte genügen, um den Nachrichtenaufwand signifikant zu verringern.

P_j bezeichne die Wahrscheinlichkeit, dass im j-ten Suchschritt der gesuchte Dienst gefunden wird und Z_j bezeichne die Anzahl der Zellen, die beim j-ten Suchschritt durchsucht werden. Das Gebiet soll in maximal q Schritten vollständig durchsucht worden sein. Wenn – wie implizit vorausgesetzt – der gesuchte Dienst auf jeden Fall im Netzwerk vorhanden ist, weil hier ja Repliken betrachtet werden, wird er spätestens im letzten Suchschritt gefunden, die Summe der Wahrscheinlichkeiten über alle Suchschritte für das erfolgreiche Finden des Dienstes muss also 1 sein. Mit

$$\sum_{j=1}^{q} P_j = 1 \qquad (4.25)$$

gilt dann für die durchschnittliche Anzahl der zu durchsuchenden Zellen

$$E_q = \sum_{j=1}^{q} P_j \cdot Z_j \tag{4.26}$$

Insbesondere gilt bei zwei Suchschritten

$$\begin{aligned} E_2 &= P_1 Z_1 + P_2 Z_2 \\ &= P_1 Z_1 + (1 - P_1) Z_2 \end{aligned} \tag{4.27}$$

Die Wahrscheinlichkeit P_1 lässt sich als Ziehen ohne Zurücklegen modellieren [Bosc06]. Die Gesamtheit der Zellen stellt dabei die Anzahl aller möglichen Kugeln dar, derjenige Anteil der Zellen, die den gesuchten Dienst mindestens einmal enthalten, ist die Anzahl der „guten“ Kugeln. Wenn r_e die Anzahl der Zellen bezeichnet, die den gesuchten Dienst mindestens einmal enthalten, dann ist die Wahrscheinlichkeit, dass in der i-ten Zelle und in allen durchsuchten Zellen davor **kein** geeigneter Dienst gefunden wird

$$\begin{aligned} P(\text{Kein Erfolg in i Zellen}) &= (1 - \frac{r_e}{n}) \cdot (1 - \frac{r_e}{n-1}) \cdot \ldots \cdot (1 - \frac{r_e}{n-(i-2)}) \cdot (1 - \frac{r_e}{n-(i-1)}) \\ &= \prod_{k=1}^{i} \left(1 - \frac{r_e}{n-(k-1)} \right) \end{aligned} \tag{4.28}$$

Damit ist die Wahrscheinlichkeit, dass in Z_1 betrachteten Zellen der gesuchte Dienst mindestens einmal verfügbar ist

$$P_1 = 1 - P(\text{Kein Erfolg in } Z_1 \text{ Zellen}) \tag{4.29}$$

Mit $P_2 = 1\text{-}P_1$ und n/n_c als Gesamtzahl der Zellen im Netzwerk gilt somit

$$\begin{aligned} E_2 &= P_1 Z_1 + P_2 Z_2 \\ &= \left(1 - \prod_{k=1}^{Z_1}\left(1 - \frac{r_e}{n-(k-1)}\right)\right) Z_1 + \left(1 - \left(1 - \prod_{k=1}^{Z_1}\left(1 - \frac{r_e}{n-(k-1)}\right)\right)\right) Z_2 \\ &= \left(1 - \prod_{k=1}^{Z_1}\left(1 - \frac{r_e}{n-(k-1)}\right)\right) Z_1 + \frac{n}{n_c} \prod_{k=1}^{Z_1}\left(1 - \frac{r_e}{n-(k-1)}\right) \\ &= Z_1 + \left(\frac{n}{n_c} - Z_1\right) \prod_{k=1}^{Z_1}\left(1 - \frac{r_e}{n-(k-1)}\right) \end{aligned} \tag{4.30}$$

Die Kosten für den ersten Suchschritt fallen also immer an. Wenn in den ersten Z_1 Zellen der gewünschte Dienst nicht gefunden wird, dann wird im zweiten Schritt das restliche Netzwerk durchsucht.

In Abb. 4-13 wird als Beispiel der Fall n=400, n_c=25, r_e= 3 und Z_1 = 1..25 dargestellt. Schon bei drei Repliken eines Dienstes ist es im betrachteten Netzwerk sinnvoll, im ersten Suchschritt nur die acht direkt benachbarten Zellen zu befragen. Für Suchen zwischen Zellen werden dann nicht mehr n/n_c-1 = 15 Zellen, sondern im Mittel nur noch weniger als 9 Zellen durchsucht, was einer Ersparnis von 40 % entspricht. Bei Vorhandensein von Repliken kann durch eine expandierende Suche also eine deutliche Kostenersparnis erzielt werden.

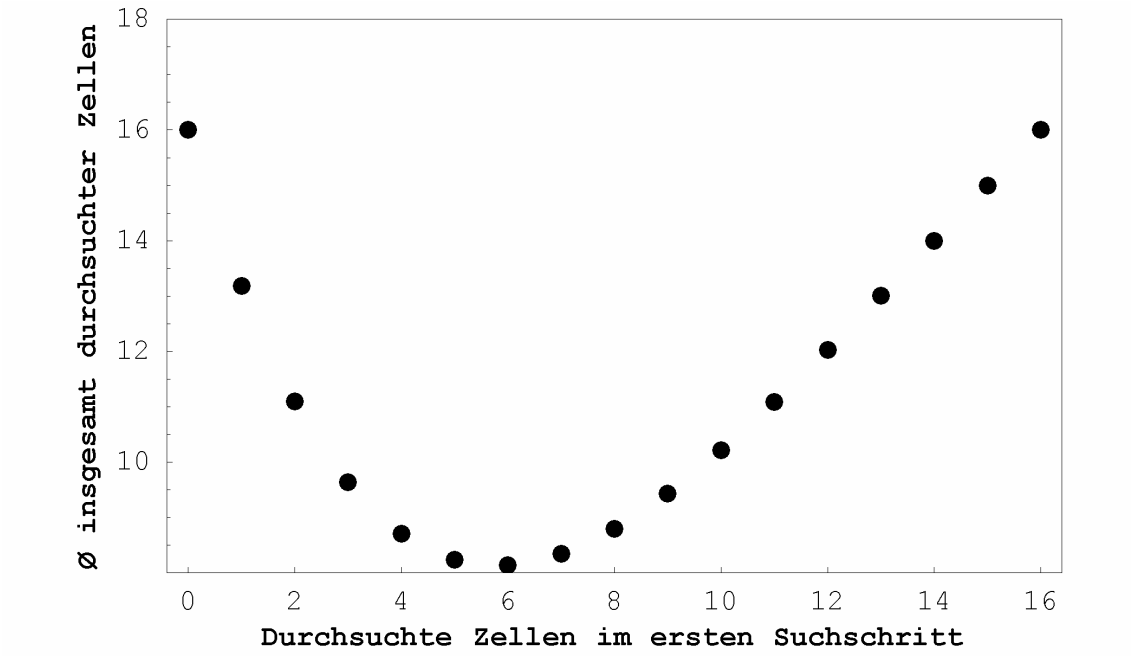

Abbildung 4-13. Wie viele Zellen bei einem Dienst mit drei Exemplaren in einem Netzwerk aus 400 Knoten und 25 Knoten pro Zelle im ersten Suchschritt durchsucht werden sollten, um die Gesamtzahl der zu durchsuchenden Zellen bei zwei Suchschritten zu minimieren.

4.5 Einsatzszenarien für CSD

Im Folgenden wird ein umfassender Vergleich von CSD mit den anderen in 2.2.2 vorgestellten Verfahren unter Berücksichtigung sowohl qualitativer wie auch quantitativer Kriterien vorgenommen (Tabelle 4-3).

Das Fluten ist eine einfach umzusetzende und sehr robuste Lösung, die sich gut für kleine Netzwerke eignet. Fluten erfüllt die meisten qualitativen Anforderungen außer der einfachen Integration und Nutzung eventuell vorhandener Infrastruktur. Hinsichtlich des Nachrichtenaufwands stellt Fluten dabei das eine Extrem dar mit sehr hohen Suchkosten, aber keinen Kosten für Bekanntmachungen. Die schon bei vergleichsweise geringen Knotenzahlen auftretenden hohen Suchkosten und die damit verbundene mangelnde Skalierbarkeit sind die größte Schwäche dieser Lösung. Durch den Einsatz von Verfahren zur effizienten Verbreitung der Suchnachrichten (2.2.2.1) und das Fluten des Operationsgebiets in mehreren Teilschritten kann hier teilweise Abhilfe geschaffen werden. Insgesamt ist das Fluten eine geeignete Wahl für kleine Netzwerke mit hoher Mobilität und/oder niedrigen Anfrageraten. In diesen Szenarien lohnen sich der Aufbau

und die Wartung einer komplexeren Organisationsstruktur nicht. Die Ziele sind stattdessen, die Kosten für Dienstbekanntmachungen zu minimieren, eine hohe Robustheit gegenüber Knotenbewegungen zu gewährleisten und eine einfache Implementierung zu ermöglichen.

CSD eignet sich für mittelgroße Netzwerke und große Netzwerke mit hoher Mobilität. CSD erfüllt wie Fluten nahezu alle qualitativen Anforderungen außer der gleichmäßigen Lastverteilung auf alle Knoten. Aufgrund der Lokalität sowohl der Suchanfragen wie auch der Bekanntmachungen kann CSD die Existenz mehrerer gleichwertiger Exemplare eines Dienstes nutzen, um die Robustheit des Systems zu erhöhen und beispielsweise auch bei einer Partitionierung des Netzwerks die Funktionsfähigkeit in den einzelnen Partitionen sicherzustellen. Gleichzeitig kann so ebenfalls die geografische Nähe von Kunden und gefundenen Dienstleistern garantiert werden. Mehrere Dienste pro Knoten werden effizient unterstützt, sie erhöhen lediglich die versendete Datenmenge, haben aber keinen Einfluss auf die Anzahl der Nachrichten. Die Hauptknoten der Zellen sind in CSD allerdings potentielle Versagenspunkte. Bei einem Ausfall eines Hauptknotens kann eine Zelle kurzzeitig nicht funktionsfähig sein. Dies hat jedoch keine Auswirkungen auf die Funktionsfähigkeit des restlichen Netzwerks. Folglich ist auch die Last nicht völlig gleich verteilt. Dies kann, muss aber kein Nachteil sein, wenn die Knoten ohnehin heterogen sind. Schließlich unterstützt CSD dynamische Erweiterungen oder Verschiebungen des Operationsgebiets, komplexe Anfragen und eine nahtlose Integration bestehender Infrastruktur, deren Knoten dann als dedizierte oder sogar untereinander verbundene Hauptknoten dienen können.
Für kleine Netzwerke übersteigt der Aufwand für den Aufbau der Organisationsstruktur von CSD die möglichen Ersparnisse gegenüber Fluten. Bei mittelgroßen Netzwerken ist CSD für alle Mobilitätsszenarien effizienter als die Vergleichsverfahren und eine geeignete Wahl. Bei großen Netzwerken ist CSD nachrichteneffizient bei hoher Mobilität der Knoten. Im Unterschied zu den anderen Protokollen minimiert CSD die Auswirkungen von Knotenbewegungen auf das Gesamtnetzwerk. Durch die Organisation in Zellen werden die Kosten für Bekanntmachungen niedrig gehalten, die Auswirkungen von Knotenbewegungen auf das Gesamtnetzwerk verringert und damit auch die Robustheit der Lösung erhöht. Bei sehr großen Netzwerken werden Dienstsuchen in CSD nachrichtenaufwendig und anderen Verfahren ist der Vorzug zu

geben. Insgesamt ist CSD also eine gute Lösung für mittelgroße und große Netzwerke insbesondere bei hoher Mobilität der Knoten.

LANES und GCLP eignen sich am besten für große, relativ statische Netzwerke. Sie werden hier aufgrund ihrer Ähnlichkeit zusammen betrachtet. Beide Lösungen können die Existenz mehrerer gleichwertiger Dienste im Netzwerk nutzen, um die Robustheit des Gesamtsystems zu erhöhen, beispielsweise um auch bei einer Partitionierung des Netzwerks funktionsfähig zu bleiben. Bei GCLP wird zusätzlich sichergestellt, dass gefundene Dienstleister in der Nähe ihrer Kunden sind. Dies ist bei LANES nicht immer der Fall, da hier zum Beispiel ein Dienstleister am Ende einer Nachbarbahn statt eines Dienstleisters auf gleicher Höhe zwei Bahnen weiter gewählt werden könnte. In beiden Verfahren werden mehrere Dienste pro Knoten effizient unterstützt, es erhöht sich lediglich die versendete Datenmenge. Die Last wird gleichmäßig über alle Knoten verteilt, so dass es zu keinen Flaschenhälsen im System kommt. Komplexe Anfragen werden ebenfalls von beiden Lösungen unterstützt. Weiterhin kann das Operationsgebiet problemlos verschoben und erweitert werden. Möglicherweise vorhandene Infrastruktur kann leichter als beim Fluten integriert werden, bringt aber aufgrund der stärkeren Dezentralisierung von LANES und GCLP geringere Vorteile als bei CSD.
Wie bei allen anderen Verfahren lohnt sich bei LANES und GCLP im Vergleich zum Fluten der Aufwand für den Aufbau der Bahnenstruktur bei kleinen Netzwerken nicht. Ein wesentlicher Nachteil von LANES und GCLP sind die Kosten, um die Konsistenz der Dienstinformation in den Bahnen aufrechtzuerhalten. Bei hoher Mobilität ist die Anzahl der erforderlichen Bekanntmachungen sehr hoch, da im Gegensatz zu CSD keine Zellstruktur verwendet wird, die eine dämpfende Wirkung ausübt. Hinzu kommt, dass die Kosten für Bekanntmachungen deutlich höher sind als bei CSD. Für mittelgroße Netzwerke ist CSD daher effizienter. Bei großen Netzwerken steigen die Kosten für Dienstsuchen unter CSD an, LANES und GCLP eignen sich dann insbesondere für große Netzwerke mit niedriger Mobilität. In diesem Szenario werden wenige Dienstbekanntmachungen, aber sehr effiziente Suchen benötigt. Für den Fall $n \to \infty$ sind die in Tabelle 4-1 angegebenen Kostenabschätzungen maßgebend und LANES und GCLP stellen (zusammen mit Rendezvous Regions) hinsichtlich des Nachrichtenaufwands die effizientesten Verfahren dar. Bei sehr großen Netzwerken wird es allerdings zunehmend schwieriger, die immer längeren Bahnen konsistent zu

halten. So müssen beispielsweise im Fall von LANES bei 5000 Knoten im Netzwerk Bahnen mit durchschnittlich 70 Knoten konsistent gehalten werden. Daher sind LANES und GCLP für derartige Netzwerke nur bei minimaler Mobilität geeignet, was dann aber z.B. eher auf Sensornetzwerke als Roboternetzwerke zutrifft. Bei einem Vergleich der beiden Verfahren untereinander ist LANES nachrichteneffizienter, während GCLP durch die Verwendung vertikaler und horizontaler Bahnen die robustere Lösung ist.

Rendezvous Regions verhält sich bezüglich des Nachrichtenaufkommens wie LANES und GCLP, weist durch die Verwendung von geografischem Hashing jedoch einige wesentliche Nachteile im praktischen Einsatz auf. Weder Suchen noch Bekanntmachungen erfolgen lokal. Einerseits ist Rendezvous Regions dadurch bei einer Partitionierung des Netzwerks nicht mehr voll funktionsfähig. Andererseits müssen im schlechtesten Fall Suchanfragen sogar für die Lokalisierung der Dienste eines direkt benachbarten Knoten beliebig weit gesendet werden (in Abhängigkeit von der Größe des Operationsgebiets). Mehrere gleiche Dienste im Netzwerk erhöhen wiederum nicht die Robustheit, weil sie alle in das gleiche Gebiet des Netzwerks abgebildet werden. Ebenso ist das Verfahren sehr ineffizient, wenn viele Dienste pro Knoten angeboten werden, die dann alle in verschiedene Teile des Netzwerks abgebildet werden. Die Last ist bei Rendezvous Regions nicht völlig gleich verteilt, da in jeder Zelle Server für die Beantwortung der Suchanfragen gewählt werden. Diese stellen auch mögliche Versagenspunkte dar, bei deren Ausfall Dienste, die in diese Zelle abgebildet wurden, nicht mehr gefunden werden können. Weiterhin sind wegen der Verwendung der Hashfunktion komplexe Anfragen entweder gar nicht oder nur unter großem Aufwand realisierbar und die Größe und Lage des Operationsgebiets muss vorher festgelegt werden. Die Integration von Infrastruktur ist möglich, indem etwa besonders leistungsfähige, statische Knoten die Aufgabe als Server übernehmen. Die Vorteile wären allerdings ohne Änderung des Protokolls geringer als bei CSD, da untereinander verbundene Server verschiedener Zellen nicht für die Verbreitung von Suchnachrichten genutzt werden könnten, weil suchende Knoten in Rendezvous Regions ihre Suchnachrichten direkt in die entsprechenden Zellen senden.

Rendezvous Regions hat relativ effiziente Bekanntmachungs- und Suchmechanismen. Bei kleinen und mittelgroßen Netzwerken schlägt sich das Fluten der Bekanntmachungen in den jeweiligen Zielregionen jedoch im Nachrichtenaufkommen nieder und das einfache Fluten bzw. CSD sind hier effizienter. Bei großen Netzwerken ist Rendezvous Regions bei niedriger Mobilität etwa genauso effizient wie LANES und

GCLP. Für Netzwerke mit hoher Mobilität sind Bekanntmachungen deutlich teurer als bei CSD und CSD ist effizienter. Wenn sehr große Netzwerke betrachtet werden gilt wie bei LANES und GCLP, dass nur nahezu statische Netzwerke effizient unterstützt werden.

Tabelle 4-3. Eine Übersicht der Verfahren zur Dienst-Entdeckung und ihrer Eigenschaften. Ein Plus bedeutet die Anforderung wird erfüllt, eine Null bedeutet die Anforderung wird teilweise erfüllt, ein Minus bedeutet die Anforderung wird nicht erfüllt.

	Fluten[3]	**CSD**	**LANES**	**GCLP**	**Rendezvous Regions**
Nutzt Repliken, um geografische Nähe von Kunden und Dienstleistern sicherzustellen und Robustheit des Gesamtsystems zu erhöhen	+	+	0	+	-
Unterstützt effizient mehrere Dienste pro Knoten	+	+	+	+	-
Last ist verteilt, kein einzelner kritischer Versagenspunkt	+	0	+	+	0
Unterstützt komplexe Anfragen	+	+	+	+	-
Erweiterung / Verschiebung des Operationsgebiets möglich	+	+	+	+	-

[3] In dieser Übersicht wird im Unterschied zu Tabelle 4-1 Fluten an Stelle von GLS aufgeführt. Bei der Betrachtung der Kommunikations- und Speicherkomplexitäten in Tabelle 4-1 sind die Werte für das Fluten trivial und deshalb wurde stattdessen GLS als ein Beispiel für ein Verfahren basierend auf verteilten Hashtabellen aufgenommen. Bei dem hier vorgenommenen qualitativen und quantitativen Vergleich ist das Fluten dagegen ein anerkanntes Verfahren. Die Betrachtung von GLS wäre hier wiederum ohne Verschulden seiner Entwickler ungünstig für GLS, weil das Verfahren speziell zur Lokalisierung von Knoten und nicht von Diensten in Netzwerken entwickelt wurde (siehe 2.2.2.2).

Einfache Integration von Infrastruktur	-	+	0	0	0
Eignung für kleine Netzwerke (Größenordnung n<100)	+	-	-	-	-
Eignung für mittelgroße Netzwerke (100<n<1000)	-	+	-	-	-
Eignung für große Netzwerke (1000<n<5000)	-	+ (hohe Mobilität)	+ (niedrige Mobilität)	+ (niedrige Mobilität)	+ (niedrige Mobilität)
Eignung für sehr große Netzwerke (n>5000)	-	-	+ (statische Netzwerke)	+ (statische Netzwerke)	+ (statische Netzwerke)

4.6 Zusammenfassung

In diesem Kapitel wurde das Cell-based Service Discovery (CSD) - Protokoll im Detail vorgestellt. CSD basiert auf einer gitterartigen Zellstruktur mit Hauptknoten in jeder Zelle, die die anderen Knoten ihrer Zellen verwalten. Suchanfragen werden an den Hauptknoten gesendet, der sie gegebenenfalls an andere Zellen des Netzwerks weiterleitet. In Netzwerken, in denen mehrere gleichwertige Dienste auftreten können, kann durch eine expandierende Suche der Nachrichtenaufwand signifikant reduziert werden. Als mögliche Erweiterungen des Grundverfahrens wurden ein Cache-System und die Verwendung einer aktiven Zwischenschicht betrachtet, in der direkt benachbarte Zellen schon im Voraus Informationen austauschen. Weiterhin wurden die Auswirkungen verschiedener Fehlerszenarien auf CSD untersucht und mögliche Lösungen diskutiert. Fehler in der Positionsbestimmung der Roboter haben, soweit sie klein im Vergleich zur Sendereichweite bleiben, keine Auswirkungen auf CSD. Hindernisse, die die Kommunikationsverbindungen stören, erhöhen den

Nachrichtenaufwand von CSD, haben aber keine Auswirkungen auf seine Funktionsfähigkeit.

Im zweiten Teil des Kapitels wurde eine Analyse eines vereinfachten Modells von CSD und ein Vergleich mit den anderen in 2.2.2 vorgestellten Verfahren durchgeführt. Weiterhin wurden die Auswirkungen von Repliken auf CSD betrachtet und die besten Einsatzszenarien für die verschiedenen Verfahren diskutiert. CSD eignet sich insbesondere für Szenarien in denen eine skalierbare Lösung für Netzwerke mit mittlerer bis hoher Mobilität der Knoten benötigt wird. Bei Existenz von Repliken wird in CSD durch eine expandierende Suche sichergestellt, dass Dienste in der Nähe effizient lokalisiert, jedoch nötigenfalls im gesamten Netzwerk nach selteneren Diensten gesucht werden kann.

5. Simulation

In diesem Kapitel wird CSD in einem Netzwerksimulator implementiert und analysiert. Im Gegensatz zur Analyse in Kapitel 4 werden nun Interferenzen bei der Kommunikation und die Mobilität der Knoten berücksichtigt. Die Implementierung im Netzwerksimulator erlaubt es, das Protokoll unter möglichst realistischen Bedingungen mit den vorgesehenen Netzwerkgrößen zu testen. Als Netzwerksimulator wird ns-2 verwendet, der die Simulation mobiler Ad-Hoc-Netzwerke unterstützt.

5.1 ns-2 Netzwerksimulator

ns-2 (*Network Simulator* v2) ist ein verbreiteter Netzwerksimulator, der die Simulation mobiler drahtloser Netzwerke unterstützt [MF]. In dieser Arbeit wurde ns-2 in der Version 2.29 verwendet. ns-2 stellt für die Simulation drahtloser Netzwerke verschiedene Modelle für die Signalausbreitung und eine Reihe unterschiedlicher Standards und Protokolle für die einzelnen ISO/OSI-Netzwerkschichten zur Verfügung. Netzwerkknoten können sich frei in einem vordefinierten Operationsgebiet bewegen. Die Knotenbewegungen werden in einer Szenario-Datei unter Angabe von Geschwindigkeit, Richtung und Pausezeiten vorgegeben. Die Behandlung von Hindernissen wird durch das ns-2 Simulationspaket jedoch nicht abgedeckt. In [JBAS05] wurde eine Erweiterung für die Integration von Hindernissen in ns-2 vorgestellt, die allerdings noch nicht validiert wurde.
Der Kern des Simulators wie auch die Verarbeitung von Nachrichtenpaketen werden in C++ beschrieben. Die Konfigurierung und die Steuerung des Ablaufs von Simulationsexperimenten erfolgt dagegen in *OTcl*, einer objektorientierten Variante der Skriptsprache *Tcl*. Diese Aufteilung dient dazu, eine hohe Performanz der Simulation sicherzustellen und gleichzeitig eine einfache und schnelle Konfigurierung von Simulationsexperimenten ohne Neukompilierung zu ermöglichen.

An dieser Stelle scheint es angebracht, eine kurze Anmerkung zur Aussagekraft von Simulationsergebnissen aus Netzwerksimulatoren zu machen. Die Übereinstimmung der Messergebnisse von Netzwerksimulatoren mit realen Messungen kann sehr hoch

sein, dies gilt insbesondere für Simulationen von Szenarien im Freien. Aus praktischen Gründen wurde dies bisher allerdings nur für eher kleine Anzahlen von Knoten und einfache Szenarien gezeigt [LBCL+01, LYNG+04]. Es wäre zu viel erwartet, von Netzwerksimulatoren exakte Aussagen über die reale Performanz zu erhalten, und daher auch wenig sinnvoll, ein Protokoll im Simulator hochgradig zu optimieren oder Protokolle im Detail miteinander vergleichen zu wollen [TMB01]. Für eine erste Einschätzung des Protokolls können die Ergebnisse jedoch ausreichen. Die Verwendung eines Netzwerksimulators stellt damit einen Kompromiss zwischen Realitätsgenauigkeit und Umsetzbarkeit dar und kann detailliertere Untersuchungen ermöglichen als eine rein mathematische Analyse.

5.2 Implementierung

Für die Simulation wurde das in Kapitel 4 vorgestellte Cell-based Service Discovery-Protokoll implementiert. Als Grundlage wird das positionsbasierte Routing-Protokoll GPSR verwendet (siehe 2.1) [KK00]. Hierfür wurden zunächst Teile von GPSR modifiziert und an die Erfordernisse von CSD angepasst. Insbesondere wurden den regelmäßig zwischen den Knoten ausgetauschten Funkfeuer-Nachrichten Informationen über die Hauptknoten der jeweiligen Zellen hinzugefügt. Diese modifizierte Version von GPSR wird dann als zugrunde liegendes Routing-Protokoll verwendet, um Nachrichten zu übermitteln, die im Rahmen des CSD-Protokolls erzeugt und ausgetauscht werden.

Weiterhin wurde für die Implementierung eine Anpassung des *Address Resolution Protocols* (ARP) vorgenommen. Ein Grund, weshalb Pakete in drahtlosen Ad-hoc-Netzwerken verloren gehen, ist das Fehlschlagen der ARP-Anfrage [NG05]. Das ARP-Protokoll ermöglicht bei Verwendung des IP-Protokolls eine Zuordnung der veränderlichen IP-Adressen zu den gerätegebundenen, festen MAC-Adressen (Media Access Control). Diese Zuordnung wird beispielsweise benötigt, wenn ein Datenpaket im lokalen Netzwerk an den Zielknoten ausgeliefert werden soll. Ein Knoten, der solch ein Datenpaket ausliefern möchte, verschickt zuerst ein ARP-Paket per Broadcast, das die IP-Adresse des Zielknotens enthält. Das Ziel selbst oder ein anderer Knoten antworten darauf mit der MAC-Adresse des Zielknotens. In mobilen Ad-hoc-

Netzwerken mit einer großen Anzahl von Knoten bindet dieser Prozess viele Ressourcen. Einerseits ist das Netzwerk sehr dynamisch, die Nachbarschaft der Knoten ändert sich regelmäßig, andererseits ist die Bandbreite vergleichsweise beschränkt. ARP-Anfragen und -Antworten teilen sich mit anderen Nachrichten das drahtlose Kommunikationsmedium und erhöhen damit die Wahrscheinlichkeit von Kollisionen und Paketverlusten. Bei den in [NG05] untersuchten Szenarien konnten über 20 % der Paketverluste auf fehlgeschlagene ARP-Anfragen zurückgeführt werden. In derselben Arbeit werden verschiedene Lösungsansätze vorgeschlagen, um die kostenintensiven ARP-Anfragen zu reduzieren. So kann beispielsweise für jedes empfangene Paket ein Austausch zwischen dem MAC-Protokoll und dem ARP-Protokoll erfolgen und somit die MAC-Adressen der direkten Nachbarn in Erfahrung gebracht werden. Schon diese Maßnahme reicht aus, um die Anzahl der Paketverluste deutlich zu senken. In der Simulation wurden deshalb ARP-Anfragen deaktiviert.

5.3 Simulationsparameter

Im Folgenden werden die Standard-Parameter für die Simulationen festgelegt und Erläuterungen zu den gewählten Wertebereichen gegeben. Wenn bei einer Simulation nicht anders angegeben wurden diese Standard-Parameter verwendet.

A. Allgemeine Simulationsparameter

ns-2 stellt verschiedene Modelle für die Signalausbreitung zur Verfügung. In den Simulationen wird das *Two Ray Ground*-Modell verwendet, das Bodenreflexionen berücksichtigt. Als Kommunikationsstandard wird IEEE 802.11b mit einer Datenrate von 11 Mbit/s verwendet. Das Intervall für den Austausch der Funkfeuer für das positionsbasierte Routing wird auf 2s gesetzt, der Mittelwert aus dem Zeitintervall von 1 bis 3 Sekunden, das die Autoren von GPSR für ihre Experimente verwendet haben [KK00]. Die Simulationszeit beträgt stets 160 Sekunden. Die ersten 60 Sekunden werden für die Initialisierung des Netzwerks genutzt und bei den Auswertungen nicht berücksichtigt.

B. Anzahl der Knoten, Operationsgebiet, Zellgröße

Es werden Netzwerke mit 100, 200 und 400 Knoten simuliert. Die Knoten operieren dabei auf Gebieten der Größen 300m x 300m oder 400m x 400m. Wenn man die Roboter also regelmäßig über die Operationsgebiete verteilen würde, würden jedem Roboter Flächen von 20m x 20m bis 30m x 30m zugewiesen. Die Größe einer Zelle wird auf 100m x 100m festgelegt. Die Operationsgebiete bestehen somit aus 9 bzw. 16 Zellen. Betrachtet werden die Verteilungen 100 Knoten in 9 Zellen, 200 Knoten in 9 Zellen, 200 Knoten in 16 Zellen sowie 400 Knoten in 16 Zellen. Hieraus ergeben sich Knotendichten von $400m^2$/Knoten bis $900m^2$/Knoten. Pro Zelle halten sich demnach im Mittel etwa 11 bis 25 Knoten auf. Hindernisse werden in der Simulation nicht betrachtet. Es wird angenommen, dass sich die Roboter im freien Feld bewegen.

C. Bewegungsmuster

Die Bewegungen der Roboter werden auf Basis des *Random Waypoint*-Modells generiert. Dabei wählen Roboter einen zufälligen Zielpunkt innerhalb des Operationsgebiets und fahren diesen direkt mit einer zufällig gewählten Geschwindigkeit an. Am Zielort wird eine Pause zufälliger Länge eingelegt, dann beginnt der Ablauf von neuem. Die Bewegungsgeschwindigkeiten der Roboter werden als gleichverteilt angenommen und betragen zwischen 0,1 m/s – 10 m/s (0,36 km/h – 36 km/h). Die Pausezeiten sind ebenfalls gleichverteilt bei Mittelwerten von 0s über 200s bis 400s.

D. Sendereichweiten

Der Senderradius wird je nach Knotendichte von 60m - 90m variiert, es wird also davon ausgegangen, dass die eingesetzten Kommunikationsgeräte über Möglichkeiten zur adaptiven Anpassung ihrer Sendereichweiten verfügen. Dies ist allerdings keine Voraussetzung für den Einsatz von CSD, sondern dient lediglich der effizienten Nutzung der zur Verfügung stehenden Übertragungsbandbreite. Es wird zu Beginn der

Simulation ein möglichst kleiner Senderradius gewählt - jedoch so, dass die Konnektivität mit hoher Wahrscheinlichkeit erhalten bleibt. In [KK00] wird gezeigt, dass in statischen Netzwerken bei Verwendung von positionsbasiertem Routing und einer

$$\text{Sendereichweite} \geq k_r \cdot \sqrt{\text{Operationsfläche} / \text{Anzahl der Knoten}} \qquad (5.1)$$

für k_r>2.5 mit hoher Wahrscheinlichkeit der Greedy-Algorithmus (siehe 2.1) zu einer erfolgreichen Nachrichtenübermittlung führt. In der Simulation wird $k_r = 3$ verwendet. Daraus ergeben sich die oben genannten Senderradien. Diese Werte sind auch unabhängig von einer Betrachtung der Konnektivität eine realistische Wahl, da hiermit die Kommunikationsreichweiten etwa zwei Größenordnungen höher als die Dimension der Roboter (1m) sind. Bei nur einer Größenordnung Unterschied wäre die maximale Kommunikationsreichweite noch im Nahbereich der Roboter, bei mehr als zwei Größenordnungen Unterschied wären die Distanzen und damit die Kosten für einen Austausch zwischen den Kommunikationspartnern - insbesondere für physische Interaktionen - sehr hoch.
Die Störreichweite von Sendern liegt etwa beim Doppelten der Sendereichweite. Bis zu dieser Entfernung können in der Simulation Sender andere Kommunikationsverbindungen stören.

E. Nachrichtengrößen

Die Nachrichtengröße für eine Dienstbeschreibung bei einer Dienstsuche wird auf 512 Bytes gesetzt. Bei einer Bekanntmachung fallen für die Beschreibungen der Dienste 4096 Bytes an (bei 8 Diensten pro Roboter, siehe F.). Derzeit gibt es kaum Erfahrungswerte bezüglich der Größe von Dienstbeschreibungen in großen heterogenen Multi-Roboter-Systemen. Erst in den letzten Jahren begannen erste Ansätze zur Erstellung umfassender Ontologien, beispielsweise für Such- und Rettungsmissionen [SM05]. Aus diesem Grund werden als Anhaltspunkt für die in der Simulation zu verwendenden Nachrichtengrößen Dienstbeschreibungen von Web-Services aus dem Internet herangezogen (siehe A.2).

F. Anzahl und Verteilung der Dienste

Jeder Roboter bietet 8 Dienste im Netzwerk an. Es sind verschiedene Verteilungen der Häufigkeiten von Diensten im Netzwerk denkbar. So können die meisten Dienste selten im Netzwerk vorkommen und nur wenige Dienste oft im Netzwerk zur Verfügung stehen. Das bedeutet, das Netzwerk besteht aus sehr heterogenen Robotern und ist als Gesamtheit sehr flexibel in der Aufgabenbearbeitung. Für jede Aufgabe stehen aber jeweils nur wenige Roboter zur Verfügung. Im umgekehrten Szenario besteht das Netzwerk eher aus gleichartigen Robotern, die meisten Dienste kommen oft im Netzwerk vor. Das Netzwerk in seiner Gesamtheit und seine Individuen sind auf bestimmte Aufgaben spezialisiert. Schließlich ist eine Verteilung denkbar, bei der die Mehrheit der Dienste in mittlerer Häufigkeit vorkommen. Soweit nicht anders erwähnt, wird für Messungen eine Verteilung wie im ersten skizzierten Fall für die Häufigkeit von Diensten gewählt. Dies stellt das für CSD herausforderndste Szenario dar. Die Dienste jedes Roboters werden dabei nach dem in Anhang A.3 gegebenen Algorithmus bestimmt. Eine typische Verteilung für 100 Knoten bei 8 Diensten pro Knoten ist in Abbildung 5-1 zu sehen.

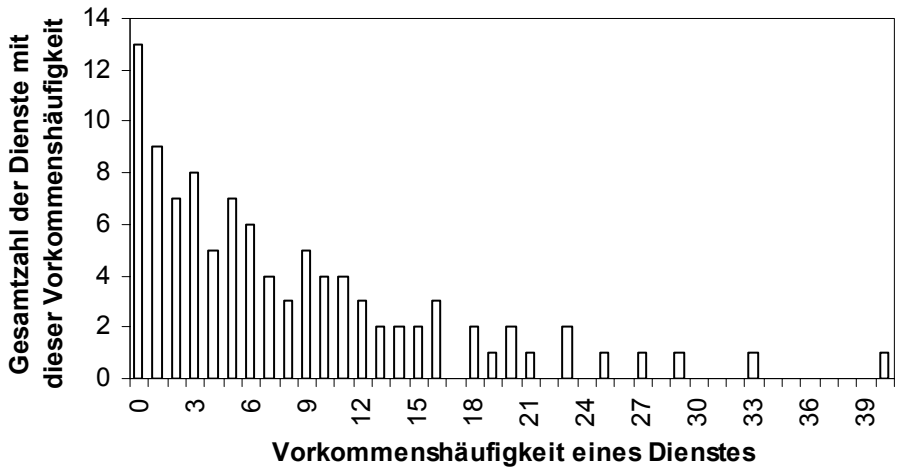

Abbildung 5-1. Eine mögliche Verteilung von Diensten im Netzwerk bei 100 Knoten und 8 Diensten pro Knoten. 13 Dienste kommen kein einziges Mal im Netzwerk vor, 9 Dienste kommen genau einmal im Netzwerk vor, usw.

G. Anfragemuster

Das Suchgebiet ist das gesamte Operationsgebiet. Die Auswahl der Dienste für Suchanfragen erfolgt zufällig mit gleicher Wahrscheinlichkeit, unabhängig von der Anzahl der Repliken eines Dienstes im Netzwerk. Dies kann als eher herausfordernde Annahme gesehen werden, da die Nachfrage nach Diensten eher entsprechend der Häufigkeit der Dienste verteilt sein müsste. Wenn dies nicht der Fall wäre, wäre das System langfristig nicht im Gleichgewicht. Die Anfragerate wird ebenfalls zufällig bestimmt und ist gleichverteilt mit einem Mittelwert von 1 Anfrage/60s. Eine Herleitung für die Anfragerate findet sich in A.4. Anfragen werden zwischen der 60. Sekunde und der 150. Sekunde versendet. Die Zeit vor den ersten Anfragen wird genutzt, um das Netzwerk zu initialisieren. Die verbleibende Zeit von der 150. Sekunde bis zum Ende der Simulation wird dazu genutzt, um in Bearbeitung befindliche Anfragen abzuschließen.

H. Expandierende Suche

Da in den hier betrachteten Szenarien das Operationsgebiet nur genauso groß ist wie das Suchgebiet, können die Nachbarzellen an die Suchanfragen versendet werden, nicht immer konzentrisch um die suchende Zelle angeordnet sein. Für Operationsgebiete aus 9 Zellen hat dies keine Auswirkungen auf die Auswahl der zu befragenden Zellen, weil ohnehin nur ein Suchschritt für die Suche zwischen Zellen vorgesehen ist. Im Fall von 16 Zellen werden im ersten Suchschritt 8 Zellen durchsucht, im zweiten Suchschritt dann die verbleibenden 7 Zellen, so dass insgesamt mit der suchenden Zelle selbst alle 16 Zellen abgedeckt sind. Die Auswahl der 8 Zellen für den ersten Suchschritt erfolgt in Abhängigkeit von der Position der suchenden Zelle. Je nachdem in welchem Viertel des Operationsgebiets sich die Zelle befindet, von der die Suche ausgeht, werden für den ersten Suchschritt 8 Nachbarnzellen so ergänzt, dass sich die suchende Zelle und die gewählten 8 Nachbarzellen einen zusammenhängenden Block aus 9 Zellen bilden, der das Ursprungsviertel der suchenden Zelle komplett abdeckt. Ist also die suchende Zelle eine der unteren linken vier Zellen des Operationsgebiets, werden zuerst die unteren linken 9 Zellen durchsucht. Für den ersten Suchschritt wird ein Zeitlimit von 4.5s

gesetzt, bei dessen Überschreitung der zweite Suchschritt gestartet und Suchanfragen an die verbleibenden 7 Zellen gesendet werden.

I. Gesamteindruck des Szenarios

Im Folgenden wird das Szenario mit 400 Knoten (und sonst unter Verwendung der Standard-Parameter) bezüglich der Gebietsgrößen, Knotengeschwindigkeiten, Sendereichweiten und Datenraten zusammenfassend beschrieben. Dies dient dazu, einen Gesamteindruck des Szenarios zu vermitteln und eine Einschätzung ihrer Wirklichkeitsnähe zu erlauben. In einem zweiten Schritt wird dann eine Skalierung der Werte auf Roboter in der Größenordnung von Khepera-Robotern (siehe 6.1.1) vorgenommen.
Es wird angenommen, dass der Durchmesser der Roboter in der Größenordnung 1m liegt. Die Größe der Operationsfläche beträgt 400m x 400m bei einer Zellgröße von 100m x 100m. Die Anzahl der Roboter beträgt 400, so dass jedem Roboter bei einer gleichmäßigen Verteilung über die Operationsfläche ein Gebiet von 20m x 20m zur Verfügung steht. Der Senderadius beträgt 60m. Nach Gl. (5.1) reicht diese Sendereichweite (gerade) aus, um die Konnektivität des Netzwerks mit hoher Wahrscheinlichkeit zu gewährleisten. Die Geschwindigkeit der Roboter ist gleichverteilt mit einem Mittelwert von 2.5 m/s bzw. 9 km/h und einem Maximalwert von 5 m/s bzw. 18 km/h. Die Dauer, um eine Zelle horizontal oder vertikal zu durchqueren beträgt damit im Mittel 40s. Für die Überbrückung einer Distanz, die dem Senderadius entspricht, werden im Mittel 24s benötigt. Bei der Wahl der Zellgröße und der Sendereichweite wurde angenommen, dass diese Werte etwa zwei Größenordnungen höher als die Ausmaße der Roboter sein sollten. Bei einem Roboter der Größe 1m erscheint beispielsweise ein Senderadius von 10m eher gering, der dadurch abgedeckte Bereich befindet sich noch im Nahfeld seiner Sensoren und er kann die Distanz schnell überbrücken. Ein Senderradius in der Größenordnung 1000m erscheint wiederum hoch. Bei physisch zu verrichtenden Arbeiten müssten die Kommunikationspartner diese erst Distanz überwinden, sie werden daher aus energetischen Gründen versuchen, einen näheren Kooperationspartner zu lokalisieren. Zudem steigt die Wahrscheinlichkeit für Interferenzen bei der Kommunikation quadratisch mit dem Senderadius. Die gleichen Überlegungen gelten auch für die Wahl

der Zellgröße. Zu geringe Zellgrößen würden ständige Zellwechsel zur Folge haben, während zu große Zellgrößen eine hohe Anzahl von Knoten pro Zelle bedeuten würde. Der letztere Fall würde dann einer zentralisierten Lösung entsprechen mit dem Hauptknoten als potentiellem Flaschenhals und Versagenspunkt. Es werden Kommunikationsmodule mit einer Datenrate von brutto 11 Mbit/s verwendet. Daraus ergeben sich nach Gl. (4.2) und Gl. (4.1)[4] bei einem Netzwerk mit 400 Knoten eine Übertragungskapazität von 0.06 KB/s bis 70 KB/s pro Knoten. In dem Zeitraum, in dem ein Knoten sich um eine Strecke weiterbewegt, die dem Senderadius entspricht, kann er also 1.44 KB bis maximal 1680 KB an Daten versenden.

Übertragen auf die Größenordnung eines Khepera-Roboters mit ca. 10 cm Durchmesser ergeben sich bei einer linearen Skalierung folgende Werte. Die betrachtete Operationsfläche entspräche einer Fläche von 40m x 40m bei einer Zellgröße von 10m x 10m. Jedem Khepera-Roboter stünde eine Fläche von 4 m^2 zur Verfügung. Bei Bewegungsgeschwindigkeiten mit einem Mittelwert von 25 cm/s dauert das Durchqueren einer Zelle 40s. Bei einer Verwendung von Bluetooth-Kommunikationsmodulen des Typs 1 mit einem Senderadius von 10m [BSIG03] dauert es 20 Sekunden, um eine Distanz zu überbrücken, die dem Senderadius entspricht. Bluetooth Version 1.X stellt eine Datenrate von ca. 700 kbit zur Verfügung und Bluetooth Version 2.0 eine Datenrate von 2,1 Mbit. Damit steht (wieder nach den Gl. (4.1) und (4.2)) jedem der 400 Roboter unter Bluetooth 1.X eine Übertragungskapazität von 4 Byte/s bis 4,5 KB/s bzw. unter Bluetooth 2.0 eine Übertragungskapazität von 12 Byte/s bis 13,5 KB/s zur Verfügung. In dem Zeitraum, in dem ein Roboter sich um eine Strecke weiterbewegt, die dem Senderadius entspricht, kann er also 80 Byte bis maximal 90 KB bzw. 240 Byte bis 270 KB an Daten versenden.

5.4 Simulationsergebnisse

In diesem Unterkapitel werden die Simulationsergebnisse vorgestellt. Bei jedem untersuchten Szenario werden dabei immer vier Knotenverteilungen betrachtet, 100

[4] Gleichung (4.2) ist eine in [GGK01] durch reale Experimente bestimmte Abschätzung. Mit Gleichung (4.1) kann ein in [GK00] durch theoretische Herleitung bestimmte maximale Übertragungskapazität berechnet werden. Diese wird im optimalen Fall erreicht, wenn die Platzierung der Knoten und die Wahl der Sendereichweiten sowie Sendezeiten von außen bestimmt werden können und die Mobilität der Knoten sowie der Aufwand für Routing-Algorithmen nicht berücksichtigt werden.

Knoten in einem Gebiet von 300m x 300m mit 9 Zellen, 200 Knoten (A) in einem Gebiet von 300m x 300m mit 9 Zellen (A), 200 Knoten in einem Gebiet von 400m x 400m mit 16 Zellen (B) sowie 400 Knoten in einem Gebiet von 400m x 400m mit 16 Zellen.

5.4.1 Anzahl der Zellwechsel

In den ersten Diagrammen werden einige grundlegende Messungen durchgeführt, die für die Organisationsstruktur von CSD relevant sind. Es wird die Anzahl der Zellwechsel der Knoten für verschiedene Mittelwerte der gleichverteilten Geschwindigkeiten (von 1 km/h über 9 km/h bis 18 km/h, Abb. 5-2) und verschiedene Mittelwerte der ebenfalls gleichverteilten Pausezeiten (von 0s über 200s bis 400s, Abb. 5-3) ermittelt.

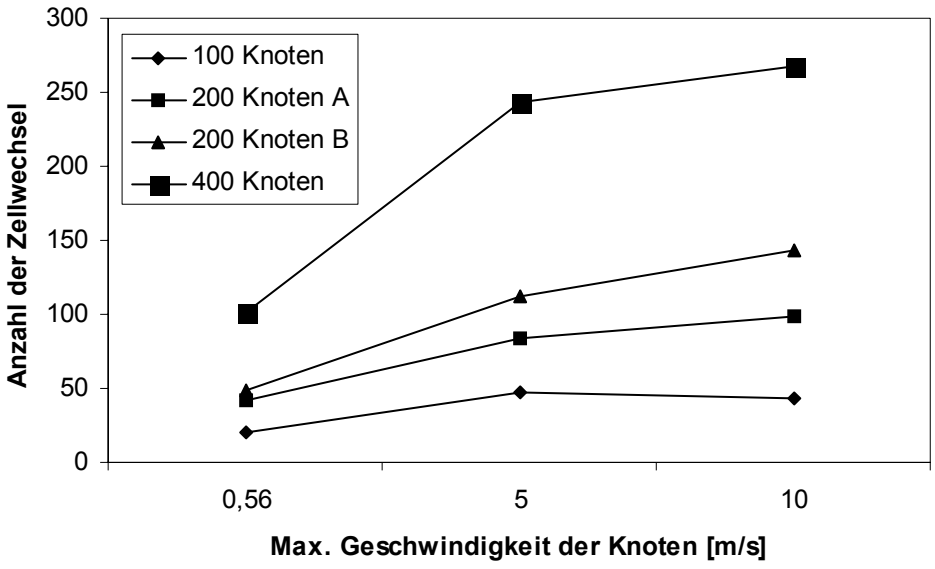

Abbildung 5-2. Einfluss der Geschwindigkeit der Knoten auf die Anzahl der Zellwechsel

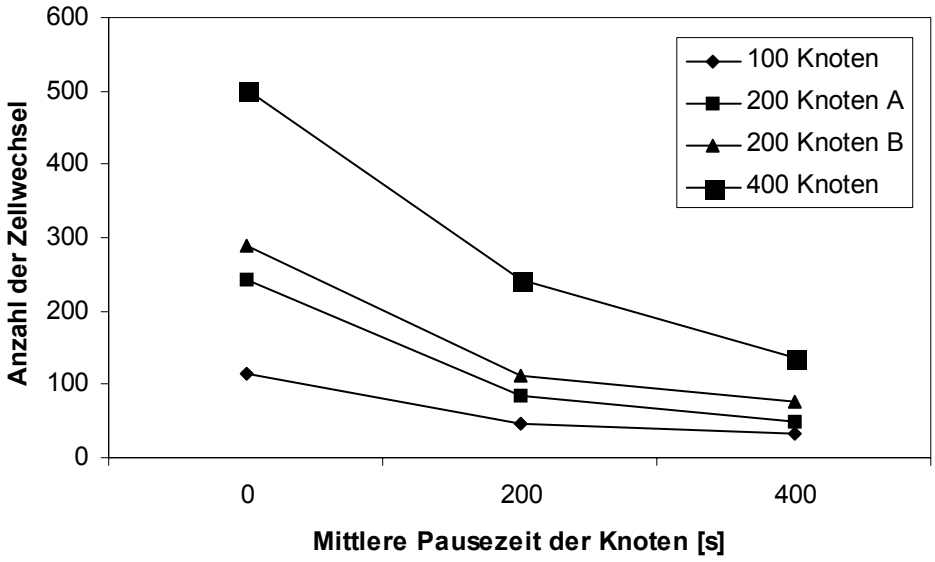

Abbildung 5-3. Einfluss der mittleren Pausezeit der Knoten auf die Anzahl der Zellwechsel

In Abbildung 5-2 sieht man, dass für sehr geringe Geschwindigkeiten nur sehr wenige Zellwechsel auftreten, weil die Knoten sich nicht schnell genug bewegen, um innerhalb der Simulationszeit die Zellgrenzen zu erreichen. Dies ist bei höheren Geschwindigkeiten nicht mehr der Fall. Hier erreichen die Knoten auch ihre Zielpunkte in der vorgegebenen Simulationszeit und die Anzahl der Zellwechsel hängt eher von der mittleren Pausezeit der Knoten ab. Man sieht in Abbildung 5-3, in der die Pausezeit variiert wird, dass für mittlere und höhere Geschwindigkeiten die Pausezeit einen stärkeren Einfluss auf die Anzahl der Zellwechsel hat als die Geschwindigkeit der Knoten.

5.4.2 Erfolgsquoten

In den folgenden Diagrammen werden die Erfolgsquoten für Suchanfragen untersucht. Die Erfolgsquote gibt den Prozentsatz der erfolgreich aufgelösten Dienstsuchen an. Neben den Geschwindigkeiten (0,36 km/h - 36 km/h) und mittleren Pausezeiten (0-400s) sind diesmal auch die Anfrageraten von Bedeutung. Die Anfragerate wird von 1 Anfrage/30s über 1 Anfrage/60s bis 1 Anfrage/90s variiert. Eine Dienst-Entdeckung gilt dann als erfolgreich, wenn der suchende Knoten eine Suchantwort mit der Identität und Position eines Dienstanbieters empfängt, der den gewünschten Dienst bereitstellt. Es wäre natürlich durchaus denkbar, dass sich der Dienstanbieter vor erfolgreicher

Kontaktaufnahme durch den suchenden Knoten so weit von seinem ursprünglichen Aufenthaltsort entfernt, dass er nicht mehr kontaktierbar ist. Die Wahrscheinlichkeit hierfür ist allerdings gering, wenn die Kontaktaufnahme direkt nach Eintreffen der Suchantwort erfolgt. Einerseits ist die Zeitdauer für Kommunikationsvorgänge im Vergleich zu der Zeitdauer für Roboterbewegungen nahezu vernachlässigbar (siehe Latenzmessungen in 5.4.4). Andererseits melden sich Dienstanbieter bei einem Zellwechsel ab und können, wenn sie sich sehr schnell bewegen, positionsabhängige Aktualisierungen an ihre Hauptknoten senden. Die Kontaktnachricht könnte also gegebenenfalls vom Hauptknoten der Zelle an den Dienstanbieter weitergeleitet werden. Zur Berechnung der Erfolgsquote werden nur Suchanfragen herangezogen, bei denen der gesuchte Dienst auch tatsächlich mindestens einmal im Netzwerk vorkommt.

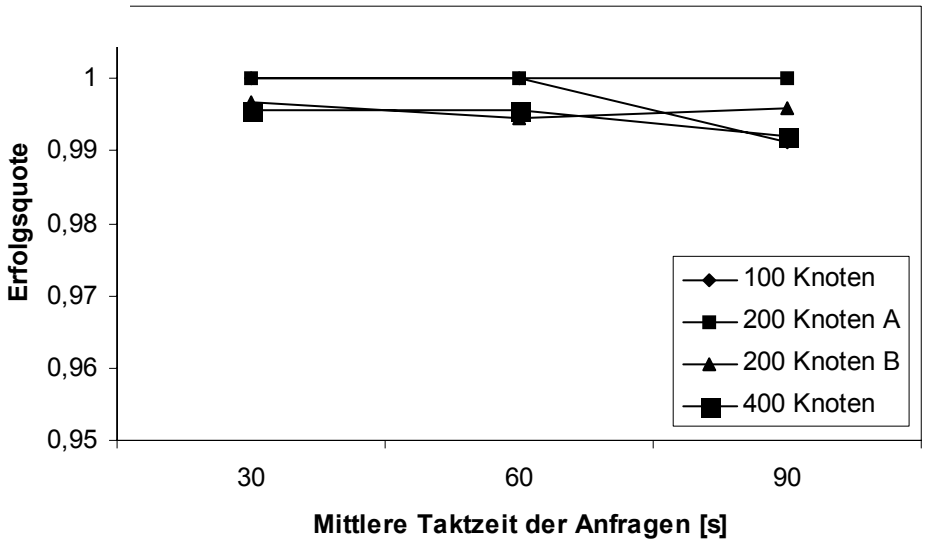

Abbildung 5-4. Einfluss der Anfragerate auf die Erfolgsquote

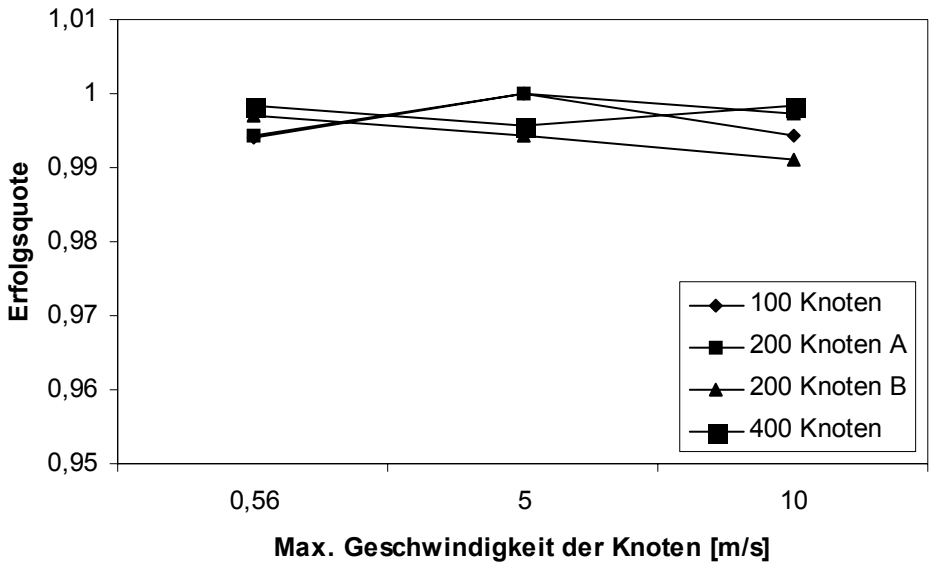

Abbildung 5-5. Einfluss der Geschwindigkeit der Knoten auf die Erfolgsquote

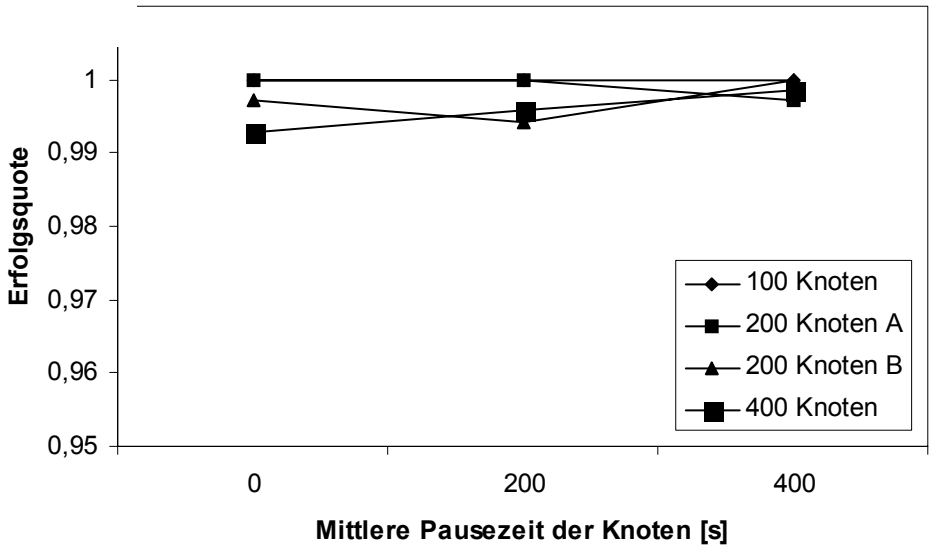

Abbildung 5-6. Einfluss der mittleren Pausezeit der Knoten auf die Erfolgsquote

Wie man in den Abb. 5-4 bis 5-6 sehen kann, beträgt die Erfolgsquote in allen betrachteten Szenarien über 99%. Die Erfolgsquote ist für die betrachteten Szenarien nahezu unabhängig von der Wahl der Simulationsparameter. Die wenigen auftretenden Fehlerfälle werden im Folgenden untersucht. Die Abweichungen in den Erfolgsquoten lassen sich auf statistische Schwankungen zurückführen.

Aus Gründen der Vollständigkeit werden hier exemplarisch für ein herausforderndes Szenario die Fehlergründe detailliert aufgeschlüsselt. Betrachtet wird das Szenario mit

400 Knoten in 16 Zellen, einer Pausezeit von 0s, Geschwindigkeiten von 0,36 km/h - 36 km/h, und einer Anfragerate von 1 Anfrage/30s. Insgesamt wurden 1143 Suchanfragen gestartet, wovon 1133 korrekt bearbeitet wurden. In 939 Fällen wird der gesuchte Dienst erfolgreich gefunden (Abb. 5-7). In 194 Fällen kann der gesuchte Dienst nicht lokalisiert werden, weil er im Netzwerk nicht existiert.

Inkorrekt bearbeitet wurden 10 Suchanfragen. Die gesuchten Dienste existierten im Netzwerk, konnten aber nicht vom suchenden Knoten lokalisiert werden. Alle diese Fehlerfälle hängen mit Zellwechsel zu kritischen Zeitpunkten zusammen. 3 Fehler traten auf, weil die suchenden Knoten ihre Zelle verlassen haben, bevor die Suche abgeschlossen war (Abb. 5-8). 4 Fehler traten auf, weil der Dienstanbieter seine Zelle im Zeitraum des Suchvorgangs verlassen hat und so für eine kurze Zeit, nach Abmeldung von der alten Zelle und vor Anmeldung in der neuen Zelle, nicht auffindbar war. 3 Fehler traten schließlich auf, weil Hauptknoten ihre Zellen während eines Suchvorgangs verlassen haben und keine Übergabe der laufenden Suchprozesse erfolgte. Möglichkeiten, wie diese Fehlerfälle einfach behoben werden können, werden in 5.5 aufgezeigt.

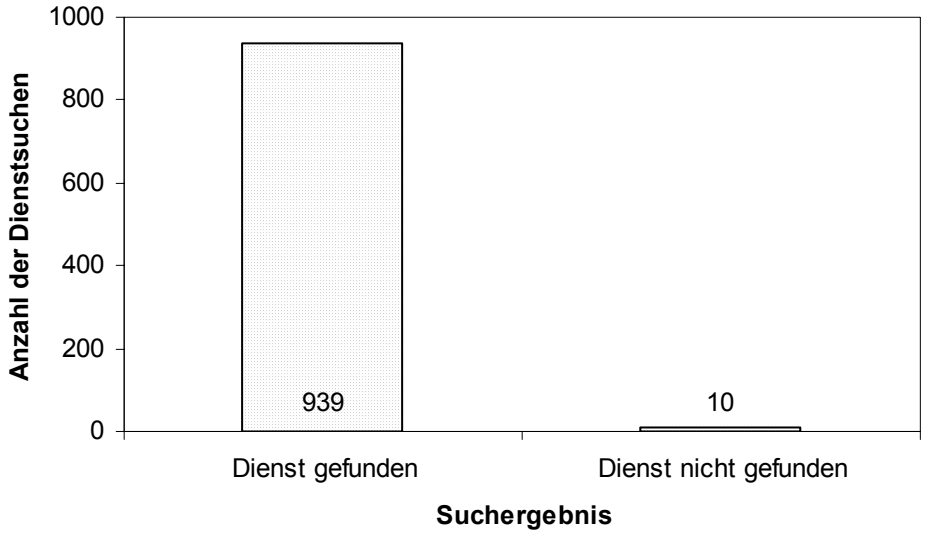

Abbildung 5-7. Anzahl der erfolgreichen und fehlgeschlagenen Dienstsuchen

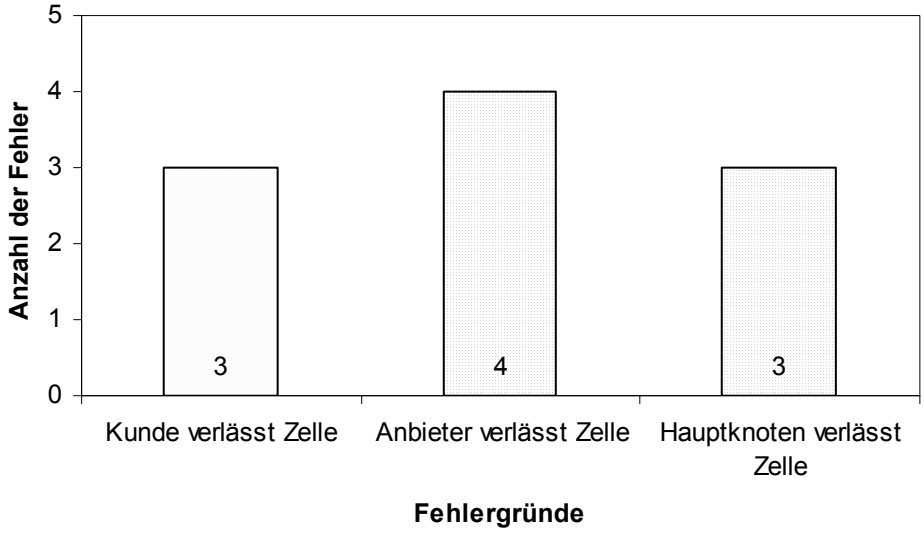

Abbildung 5-8. Fehlergründe für die fehlgeschlagenen Dienstsuchen

5.4.3 Nachrichtenaufkommen

Nun wird das Nachrichtenaufkommen gemessen. Die Parameter werden wie bei der Messung der Erfolgsquoten variiert. In Abbildung 5-9 ist dazu die mittlere Zahl der Dienstsuchen für verschiedene Anfrageraten dargestellt. Bei der Messung des Nachrichtenaufkommens wird bei Multi-Hop-Verbindungen jeder Nachrichtsprung separat gezählt.

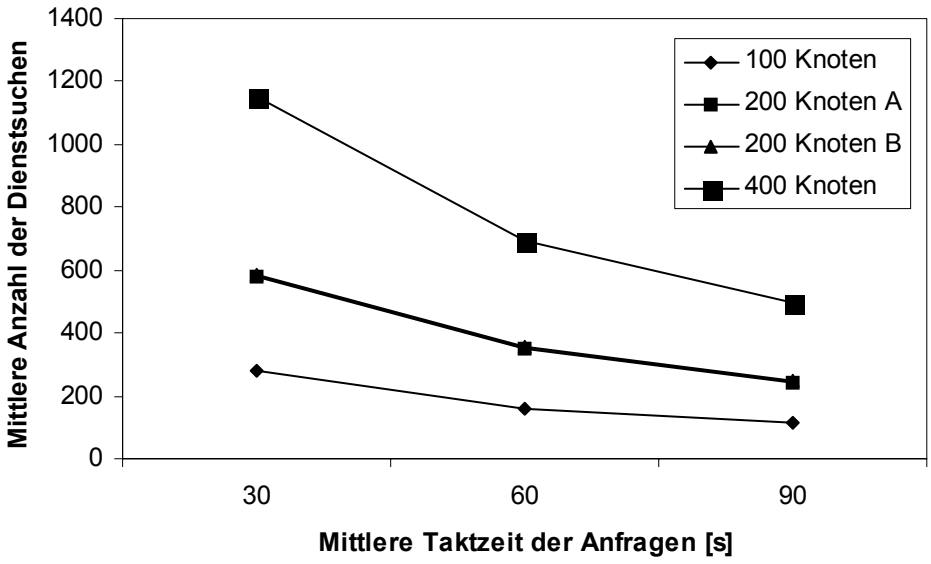

Abbildung 5-9. Anzahl der Dienstsuchen für verschiedene Anfrageraten

Zuerst wird das Nachrichtenaufkommen hinsichtlich der Anzahl der insgesamt versendeten Nachrichten sowie der Anzahl der versendeten Nachrichten pro Dienstsuche betrachtet. In den nachfolgenden Diagrammen werden dann für die Fälle 200 Knoten in 9 Zellen sowie 200 Knoten in 16 Zellen die Anzahl der versendeten Nachrichten und die versendeten Datenmengen den vier unterschiedlichen Nachrichtenklassen ‚Funkfeuer', ‚Bekanntmachung', ‚Suche in Zelle' und ‚Suche zwischen Zellen' zugeordnet. Schließlich werden die mittleren versendeten Datenmengen getrennt für gewöhnliche Knoten und für Hauptknoten dargestellt.

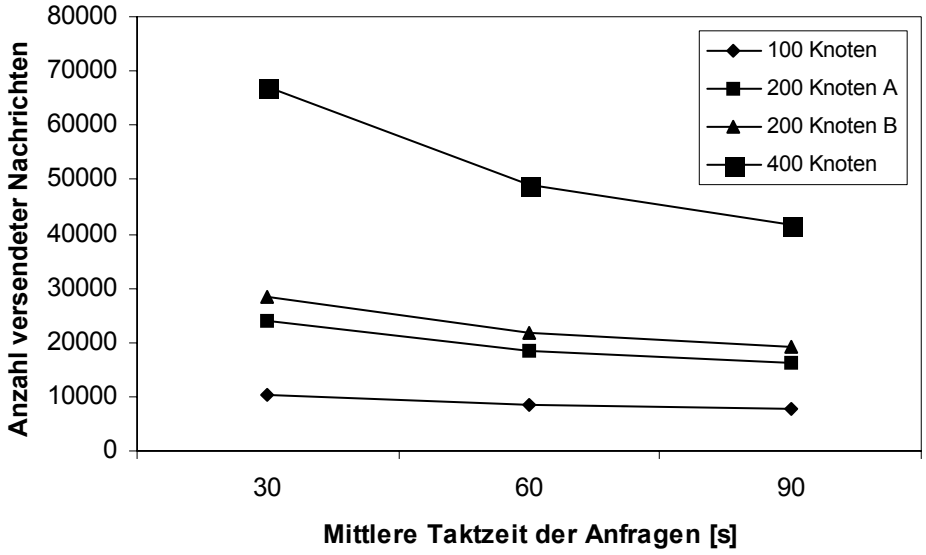

Abbildung 5-10. Einfluss der Anfragerate auf die Gesamtzahl der Nachrichten

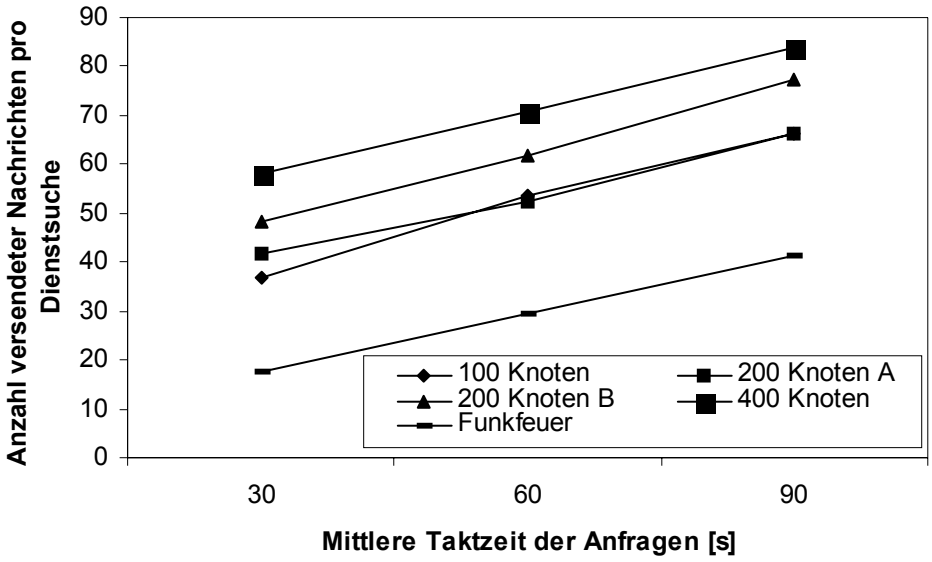

Abbildung 5-11. Einfluss der Anfragerate auf die mittlere Zahl der Nachrichten je Dienstsuche

Bei einer Variation der Anfragerate sieht man, dass bei abnehmender Anfragerate die Gesamtzahl der Nachrichten abnimmt und die mittlere Nachrichtenanzahl pro Dienstsuche zunimmt (Abb. 5-10 und 5-11). Dies ist dadurch zu erklären, dass der Aufwand für den pro-aktiven Nachrichtenaustausch wie das Versenden von Dienstaktualisierungen oder Funkfeuer unabhängig von der Anfragerate ist und gleich bleibt. Der hierfür erforderliche Aufwand verteilt sich damit bei einer sinkenden Anfragerate rechnerisch auf eine geringere Zahl von Dienstsuchen. Die Geschwindigkeit der Knoten und die Pausezeiten haben wie schon bei der Messung der Erfolgsquoten vergleichsweise geringe Auswirkungen auf die Anzahl der versendeten Nachrichten (siehe Diagramme A-2 bis A-5 im Anhang). Eine höhere Geschwindigkeit ist tendenziell mit einer geringfügig höheren Nachrichtenzahl verbunden, während höhere Pausezeiten mit leicht sinkenden Nachrichtenzahlen einhergehen.

Weiterhin fällt auf (Abb. 5-9 bis 5-11), dass der Nachrichtenaufwand überproportional mit der Anzahl der Knoten steigt. Hier nehmen zwei Faktoren Einfluss auf die Anzahl der versendeten Nachrichten: Die Sendereichweite der Knoten auf der einen Seite, die für jedes Szenario basierend auf der Knotendichte gewählt wurde, und die Anzahl der Zellen auf der anderen Seite. Eine höhere Sendereichweite verringert die durchschnittliche Anzahl der Nachrichtensprünge, während sich eine Erhöhung der Zellenanzahl gegenteilig auswirkt.

So sind für 400 Knoten in 16 Zellen die mittlere Anzahl der Nachrichtensprünge am höchsten (Abb. 5-12). Funkfeuernachrichten wurden hierbei nicht berücksichtigt, weil diese immer nur über genau einen Sprung versendet werden. Dieses Szenario hat die höchste Knotendichte und damit die niedrigste Sendereichweite, und gleichzeitig auch die größte Zellenanzahl. Bei 100x100 Knoten in 9 Zellen verhält es sich genau umgekehrt. Die Sendereichweite ist die größte aller betrachteten Szenarien und die Anzahl der Zellen ist am niedrigsten, folglich hat dieses Szenario auch die niedrigste mittlere Anzahl an Nachrichtensprüngen. In den Szenarien 200 Knoten in 9 Zellen und 200 Knoten in 16 Zellen ist die mittlere Anzahl an Nachrichtensprüngen ungefähr gleich. Hier heben sich die beiden gegensätzlich auswirkenden Effekte teilweise auf.

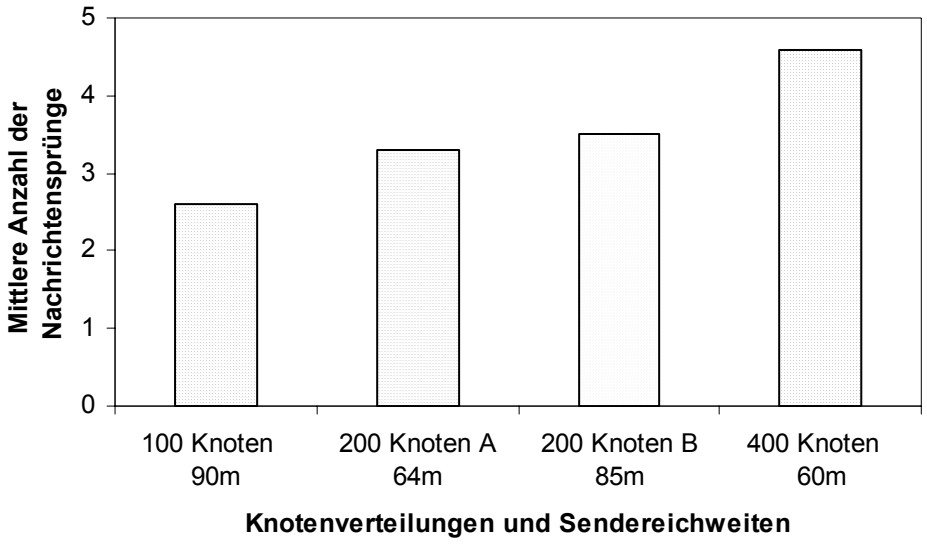

Abbildung 5-12. Mittlere Anzahl der Nachrichtensprünge für verschiedene Knotenverteilungen und Senderadien. Funkfeuernachrichten sind hier nicht berücksichtigt, da sie immer nur über genau einen Sprung versendet werden.

Im Folgenden werden daher der Fall 200 Knoten in 16 Zellen und der Fall 200 Knoten in 9 Zellen im Detail betrachtet. Hierfür wird die Zahl der versendeten Nachrichten und Datenmengen nach Nachrichtentyp aufgeschlüsselt. ‚Funkfeuer' stellen einen eigenen Nachrichtentyp dar. Sie werden für das positionsbasierte Routing benötigt und gleichzeitig für die Wahl der Hauptknoten genutzt. Unter ‚Bekanntmachungen' werden Dienstbekanntmachungen von Knoten, Abmeldungen von Knoten aus ihrer Zelle und Übergaben von Hauptknoten an Hauptknoten zusammengefasst. Dienstbekanntmachungen stellen dabei den größten Teil der Nachrichten dar. Unter

‚Suche in Zelle' werden Suchanfragen und -antworten für Dienstsuchen in einer Zelle zusammengefasst, entsprechend werden Suchanfragen und -antworten für Dienstsuchen zwischen Zellen unter ‚Suche zwischen Zellen' zusammengefasst. Die Verteilungen der Nachrichten auf die unterschiedlichen Nachrichtenklassen für 100 Knoten in 9 Zellen und 400 Knoten in 16 Zellen verhalten sich charakteristisch ähnlich zu den Verteilungen für 200 Knoten in 9 Zellen bzw. 200 Knoten in 16 Zellen und werden hier deshalb nicht aufgeführt.

In den Abbildungen 5-13 bis 5-16 ist zu sehen, dass im Fall von 200 Knoten in 16 Zellen im Vergleich zu 200 Knoten in 9 Zellen aufgrund der größeren Sendereichweite tendenziell eine leichte Abnahme der Anzahl der Nachrichtensprünge festzustellen ist. Diese Reduktion wird jedoch durch eine Zunahme des Nachrichtenaustauschs zwischen Zellen aufgehoben. Weiterhin ist zu sehen, dass Funkfeuernachrichten einen erheblichen Anteil an der Anzahl der versendeten Nachrichten, aber aufgrund ihrer geringen Größe nur einen kleinen Anteil an der insgesamt versendeten Datenmenge stellen. Umgekehrt verhält es sich bei Bekanntmachungen. Sie stellen angesichts ihrer Größe einen relativ großen Anteil an der versendeten Datenmenge, es werden aber nur vergleichsweise wenige verschickt. Die Suche innerhalb einer Zelle fällt weder hinsichtlich der Anzahl der Nachrichten noch der Datenmenge stark ins Gewicht. Am kostspieligsten ist die Suche zwischen Zellen. Einerseits müssen Nachrichten über größere Entfernungen an viele Zellen übermittelt werden, was eine hohe Nachrichtenzahl ergibt. Andererseits sind die Suchnachrichten von mittlerer Größe, dies schlägt sich in der versendeten Datenmenge nieder. Wie diese Kosten reduziert werden können, wird in 5.5 erläutert.

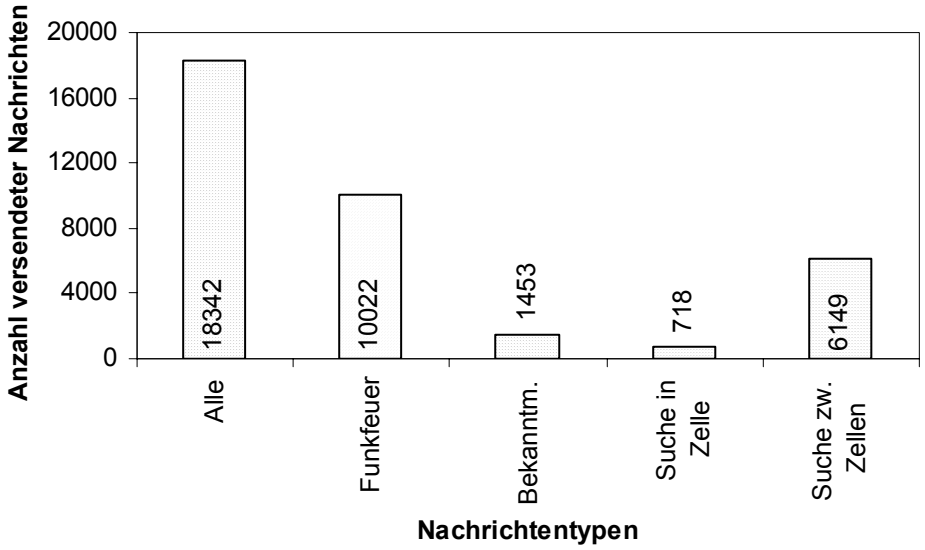

Abbildung 5-13. Aufschlüsselung der Anzahl der versendeten Nachrichten nach Typ für 200 Knoten in 9 Zellen

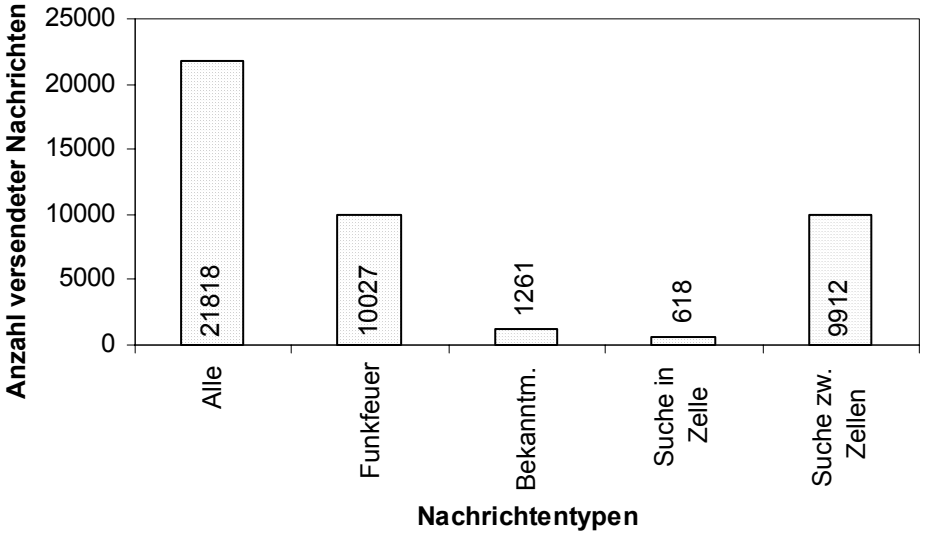

Abbildung 5-14. Aufschlüsselung der Anzahl der versendeten Nachrichten nach Typ für 200 Knoten in 16 Zellen

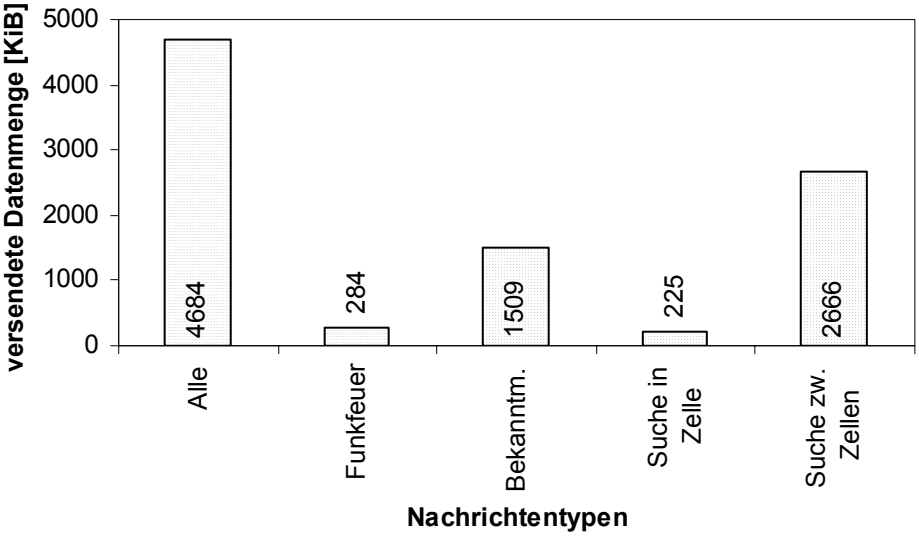

Abbildung 5-15. Aufschlüsselung der versendeten Datenmenge nach Nachrichtentyp für 200 Knoten in 9 Zellen

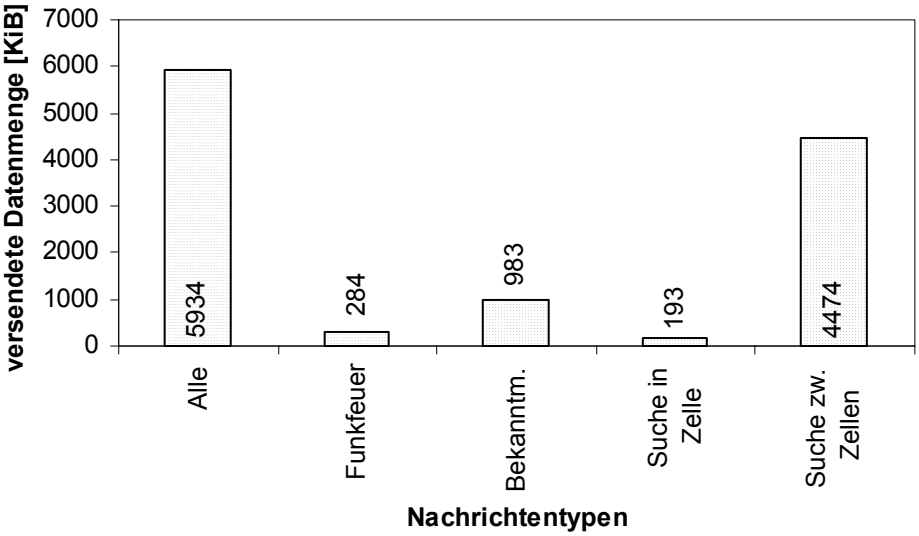

Abbildung 5-16. Aufschlüsselung der versendeten Datenmenge nach Nachrichtentyp für 200 Knoten in 16 Zellen

Bei einer Steigerung der Anfragerate auf 1 Anfrage/30s verdoppelt sich erwartungsgemäß in etwa die Anzahl der Suchnachrichten, während die Anzahl der anderen Nachrichtentypen von der Steigerung der Anfragerate unbetroffen bleibt (siehe Diagramme A-6 bis A-9 im Anhang).

Schließlich werden noch die mittleren versendeten Datenmengen getrennt für gewöhnliche Knoten und Hauptknoten betrachtet. Auch hier lässt sich feststellen, dass

die Bewegungsgeschwindigkeit der Knoten und die Pausezeiten im Gegensatz zu den Anfrageraten einen eher geringen Einfluss auf die versendeten Datenmengen haben (siehe Diagramme A-10 bis A-13 im Anhang). Tendenziell haben höhere Geschwindigkeiten bzw. Anfrageraten dabei erwartungsgemäß eine erhöhte Datenmenge und erhöhte Pausezeiten eine verringerte Datenmenge zur Folge. Ferner lässt sich aus den Diagrammen eine höhere Belastung der Hauptknoten in Feldern mit 16 Zellen ablesen (Abb. 5-17 und 5-18). So ist die Belastung der gewöhnlichen Knoten bei 200 Knoten in 9 bzw. 16 Zellen etwa gleich, die Belastung der Hauptknoten im Fall 200 Knoten in 16 Zellen im Vergleich zu 200 Knoten in 9 Zellen jedoch höher. In 5.5 wird erläutert, wie dieser Effekt weitgehend beseitigt werden kann.

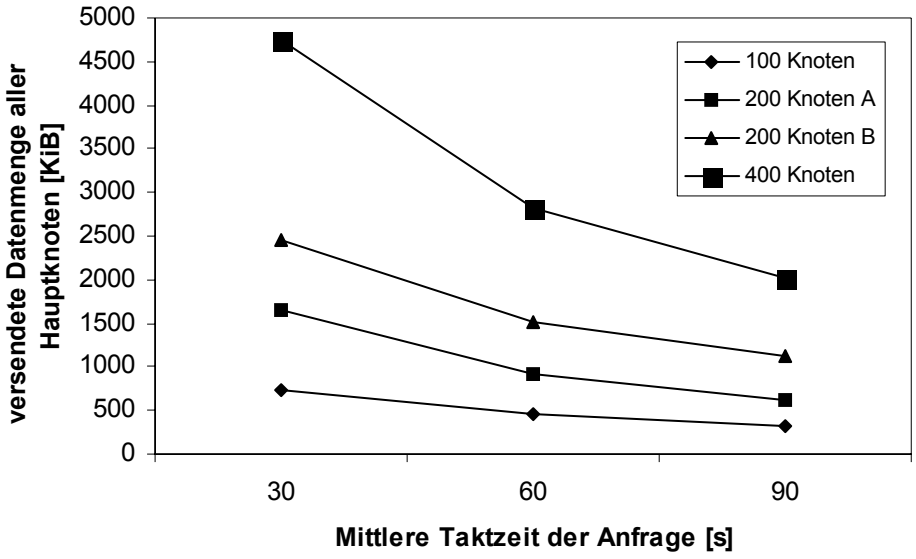

Abbildung 5-17. Einfluss der Anfragerate auf die versendete Datenmenge aller Hauptknoten

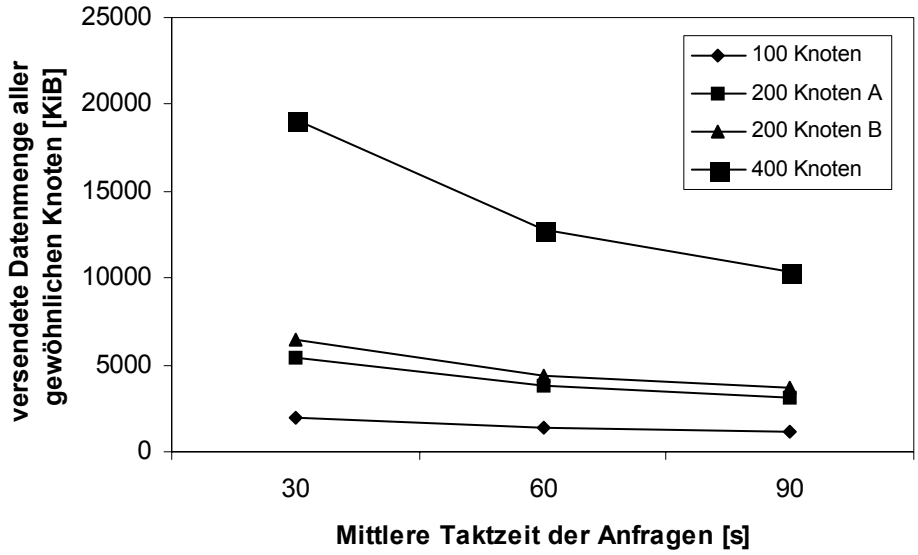

Abbildung 5-18. Einfluss der Anfragerate auf die versendete Datenmenge aller gewöhnlichen Knoten

5.4.4 Latenz

Abschließend wird die Latenz gemessen. Die Latenz ist hier der Zeitraum zwischen dem Abschicken der ersten Suchanfrage und dem Erhalt der ersten passenden Suchantwort. Die gemessenen Abweichungen der Latenzen von Szenario zu Szenario waren nur gering. In Abbildung 5-19 sind die minimalen, die maximalen und die Mittelwerte der Latenzen über alle untersuchten Szenarien (mittlere Geschwindigkeit von 1 km/h bis 18 km/h, mittlere Pausezeit von 0s bis 400s, mittlere Anfragerate von 1 Anfrage/30s bis 1 Anfrage/90s) für die vier verschiedenen Knotenverteilungen dargestellt. Der wesentliche Einflussfaktor hier ist das Zeitlimit, ab dem bei einer zwei- oder mehrstufigen expandierenden Suche der jeweils nächste Suchschritt ausgelöst wird. Damit ist auch der im Diagramm zu sehende Sprung der Latenz zwischen den Knotenverteilungen mit 9 Zellen und 16 Zellen zu erklären. Insgesamt liegen die Latenzzeiten deutlich unter den Zeiten, die beispielsweise für Roboterbewegungen notwendig sind.

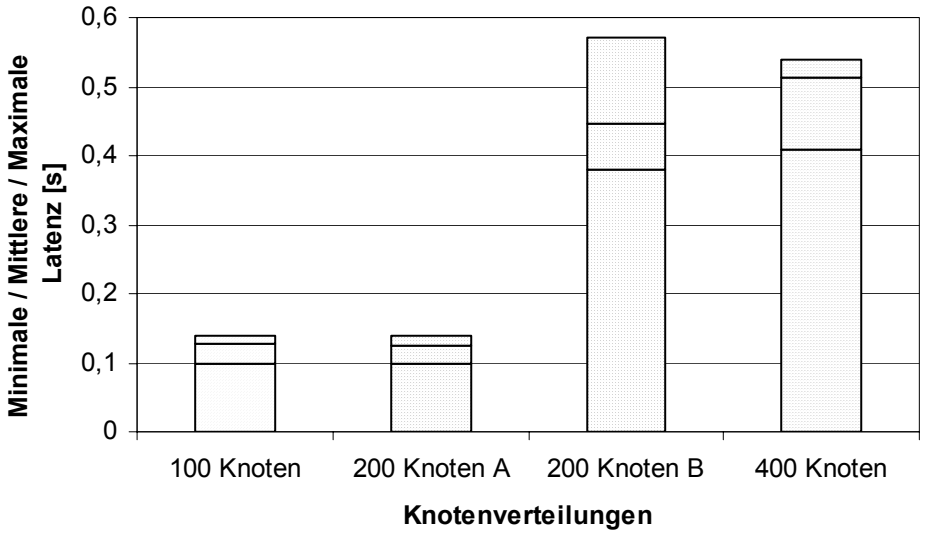

Abbildung 5-19. Latenzen für verschiedene Knotenverteilungen

5.5 Auswertung

Im Folgenden werden die Simulationsergebnisse zusammengefasst und mögliche Erweiterungen diskutiert.

CSD zeigte in den untersuchten Szenarien weitgehend unabhängig von der Wahl der Simulationsparameter eine hohe Erfolgsquote von über 99 % bei der Lokalisierung von Diensten. In einem detailliert untersuchten Szenario wurden lediglich 10 von insgesamt 1143 Dienstsuchen inkorrekt behandelt, das heißt, der gesuchte Dienst existierte im Netzwerk, konnte aber nicht lokalisiert werden. Die aufgetretenen Fehler sind alle auf Zellwechsel von Knoten zu kritischen Zeitpunkten zurückzuführen (siehe 5.4.2). Da es sich nur um wenige Fehlerfälle handelt und sie bei der Untersuchung der Performanz des Protokolls in seiner Gesamtheit kaum ins Gewicht fallen, wurden keine Mechanismen für diese speziellen Situationen implementiert. Bei einer Nutzung des Protokolls in der Praxis ließen sich die Fehler aber leicht beheben. Für den ersten beschriebenen Fehlerfall würde es genügen, wenn Hauptknoten die gewöhnlichen Knoten nach ihrer Abmeldung aus der Zelle nicht gleich vergessen würden. Da Knoten sich im Vergleich zur Geschwindigkeit der Nachrichtenübermittlung nur vergleichsweise langsam bewegen, können Nachrichten an die letzte bekannte Position

nachgeschickt werden und es besteht eine hohe Wahrscheinlichkeit, dass die Nachricht korrekt empfangen wird. Für den zweiten Fehlerfall gilt das Gleiche. Die Dienste eines Knotens, der eine Zelle verlässt, werden nicht sofort entwertet, sondern erst nach einer kurzen Zeit. Damit bleiben die Dienste auffindbar, bis sich der Knoten in seiner neuen Zelle angemeldet hat. Für den dritten Fall bedarf es schließlich lediglich einer Übergabe der laufenden Suchprozesse von den Hauptknoten an ihre Nachfolger, wenn die Hauptknoten ihre Zelle verlassen.

Das Nachrichtenaufkommen erwies sich als relativ unabhängig von der Geschwindigkeit der Knoten und den Pausezeiten und hing im Wesentlichen von der Anfragerate ab. Dies ist ein Indiz für die Robustheit des zugrunde liegenden positionsbasierten Routing-Protokolls. Tendenziell haben höhere Knotengeschwindigkeiten bzw. Anfrageraten ein erhöhtes Nachrichtenaufkommen zur Folge, während höhere Pausezeiten mit sinkenden Nachrichtenaufkommen einhergehen. Bei einer Zuordnung des Nachrichtenaufkommens zu verschiedenen Nachrichtenklassen lässt sich feststellen, dass es gerade bei Dienstbekanntmachungen gelungen ist, die Zahl der versendeten Nachrichten klein zu halten. Dies ist von Bedeutung, weil Dienstbekanntmachungen weit überdurchschnittliche Nachrichtengrößen aufweisen. Die Suche in Zellen fiel beim Nachrichtenaufkommen fast nicht ins Gewicht. Als vergleichsweise kostspielig und eine Belastung für die Hauptknoten erwies sich dagegen die Suche zwischen Zellen. Es ist jedoch zu erwarten, dass die Kosten für Suchen zwischen Zellen noch erheblich gesenkt werden können. Erstens wurde für die Auslieferung von Suchnachrichten kein positionsbasiertes Multicasting verwendet. Dabei werden gleiche Nachrichten für benachbarte Zielgebiete (wie Suchnachrichten in CSD) zu Beginn als einzelne Nachricht verschickt und erst kurz vor dem Zielgebiet kopiert und verteilt. Mit der Integration von positionsbasiertem Multicasting im Routing-Protokoll könnten die Suchkosten in den betrachteten Szenarien um schätzungsweise bis zu 50 % verringert werden (siehe Abschätzung in A.5). Zweitens wurde bei der Simulation angenommen, dass das gesamte Operationsgebiet nur genauso groß ist wie das Suchgebiet. Dies ist die für CSD schlechtere Annahme. Ebenso wäre es denkbar, dass das Operationsgebiet, in dem die Roboter stationiert sind, größer ist als das Suchgebiet. In diesem Fall wären die jeweiligen Suchgebiete für einen größeren Anteil der Zellen konzentrisch um die Zellen

verteilt, was eine Verringerung der Kommunikationsdistanz für Suchnachrichten bedeuteten würde.

Die gemessenen Latenzzeiten für Dienstsuchen betrugen 0,1s bis 0,14s für Szenarien mit 9 Zellen und 0,38s bis 0,57s für Szenarien mit 16 Zellen. Der wesentliche Einflussfaktor hier war das Zeitlimit, ab dem bei einer expandierenden Suche der jeweils nächste Suchschritt ausgelöst wird.

5.6 Zusammenfassung

In diesem Kapitel wurde CSD im Netzwerksimulator untersucht. Hierfür wurden zunächst geeignete Simulationsparameter bestimmt und dann Simulationen für 28 verschiedene Szenarien durchgeführt. CSD zeigt dabei eine sehr hohe Erfolgsrate von über 99 % beim Finden von Diensten und erweist sich hinsichtlich des Nachrichtenaufwands als sehr robust gegenüber der Mobilität der Knoten. So konnte die Anzahl der aufgrund weit überdurchschnittlicher Nachrichtengrößen teuren Dienstbekanntmachungen gering gehalten werden. Damit zeigt sich CSD erwartungsgemäß erfolgreich bei der Minimierung der Auswirkungen von Knotenbewegungen. Als nachrichtenaufwendige Komponente erwies sich die Suche zwischen Zellen. Dies kann jedoch, wie gezeigt wurde, durch die Verwendung von positionsbasiertem Multicasting und der Wahl geeigneter Operationsgebiete behoben werden. Damit erweist sich CSD als geeignet für eine große Bandbreite von Einsatzszenarien im Bereich mobiler Roboternetzwerke.

6. Implementierung im realen Multi-Roboter-System

In diesem Kapitel wird beschrieben, wie eine vereinfachte Version von CSD auf einem realen Multi-Roboter-System umgesetzt wird. Im ersten Abschnitt wird die Hardware-Plattform vorgestellt, die für die Implementierung eingesetzt wird. Danach werden die umgesetzte Lösung und die für die Implementierung erforderlichen Anpassungen beschrieben. Anschließend wird die entwickelte Lösung in einem Beispielszenario angewandt und auf diese Weise einige Vorteile servicebasierter Multi-Roboter-Systeme demonstriert. Schließlich wird im letzten Abschnitt gezeigt, wie eine Web-Service-Schnittstelle in die hier verwendete Roboterplattform integriert wird. Viele Herausforderungen in den Forschungsgebieten der serviceorientierten Softwarearchitekturen und der servicebasierten Multi-Roboter-Systeme stimmen überein. Es erscheint daher sinnvoll, die Entwicklungen im Bereich serviceorientierter Softwarearchitekturen auch in der Robotik zu nutzen.

6.1 Implementierungsplattform

In diesem Abschnitt wird kurz die für die Implementierung verwendete Hardware vorgestellt. Dies sind im Wesentlichen der *Khepera* II Roboter und die *Telewerkbank*, eine Entwicklungs- und Testplattform für Multi-Roboter-Systeme.

6.1.1 Khepera II Roboter

Als Basisplattform für die Implementierung von CSD auf einem realen Robotersystem wird der *Khepera* II Roboter verwendet (Abb. 6-1) [KTEA]. Der Khepera ist ein autonomer, mobiler Roboter der Firma K-Team. Die einfache, aber für viele Anwendungen ausreichende Grundausstattung aus einem Motorola 68331 Mikroprozessor mit 25 MHz, 512 KiB flüchtigem und 512 KiB nicht-flüchtigem Speicher, 8 Infrarotsensoren, 2 Motoren mit Inkrementalgebern und Akkus bildet eine geeignete Basis für viele Roboter-Experimente. Mit seinen kleinen Abmessungen (Durchmesser 7 cm, Höhe 3 cm) und der einfachen Handhabung eignet sich der Khepera insbesondere für Multi-Roboter-Experimente. Der Roboter kann durch

Erweiterungsmodule ausgebaut werden, die auf die Basisplattform aufgesteckt werden. Es lassen sich mehrere Module übereinander aufstecken, die dann über den so genannten K-Bus mit dem Roboter Daten austauschen können. So stehen in der Fachgruppe Schaltungstechnik beispielsweise verschiedene Kameramodule, FPGA-Module und Greifermodule zur Verfügung. Für die vorliegende Arbeit ist dabei besonders das Bluetooth-basierte Kommunikationsmodul von Bedeutung. Es basiert auf einem Funkmodul der Firma Mitsumi mit integrierter Antenne und wurde entwickelt, um eine effiziente drahtlose Kommunikation für den Khepera-Roboter zu ermöglichen. Das Kommunikationsmodul kann genutzt werden, um bis zu vier Khepera - Roboter in einem Piconetz zu organisieren [Gros04].

Die Programmierung des Khepera erfolgt in der Regel in der Programmiersprache C. Hierfür stellt K-Team die Programmierumgebung *KTProject* zur Verfügung. Eine C-Bibliothek stellt grundlegende Befehle bereit, über die der Entwickler auf das BIOS des Roboters zugreifen kann. Mit den Grundbefehlen lassen sich die Sensoren und Aktoren ansprechen und es kann auf integrierte Komponenten wie Zeitgeber sowie auf interne und externe Schnittstellen zugegriffen werden. Die in C geschriebenen Programme werden auf dem PC für den Khepera kompiliert und anschließend über eine serielle Verbindung an den Roboter übertragen. Nach der Übertragung kann das Programm entweder in den nicht-flüchtigen Speicher geschrieben oder direkt ausgeführt werden.

Abbildung 6-1. Der Khepera II Roboter mit Bluetooth-Kommunikationsmodul

6.1.2 Telewerkbank

Die *Telewerkbank* (TWB) stellt eine Entwicklungs- und Experimentierplattform für Multi-Roboter-Systeme dar, die im Fachbereich Schaltungstechnik entwickelt wurde (Abb. 6-2 und 6-3) [TWR05]. Für Experimente steht eine 2x2 m große Fläche zur Verfügung, die bei Bedarf auch in bis zu vier 1m x 1m große Felder aufgeteilt werden kann. Mit der Telewerkbank lassen sich Experimente sowohl vor Ort als auch über das Internet durchführen. Mittels Videostreaming ist es möglich, ein Experiment in Echtzeit zu beobachten. Weiterhin archiviert der Server die Versuchsdaten, so dass Versuche nachträglich analysiert und miteinander verglichen werden können.

Die für diese Arbeit wichtigste Funktion ist jedoch die Möglichkeit, Roboter auf der Versuchsfläche global zu lokalisieren. Die Orientierung der Roboter in Grad und ihre x/y-Koordinaten zu einem festen Bezugspunkt lassen sich millimetergenau bestimmen. Hierzu dienen über der Versuchsfläche angebrachte Kameras. Außerdem wird auf jedem Roboter eine eindeutige farbliche Markierung befestigt. Ein zentraler Server wertet die anfallenden Bilddaten aus, berechnet daraus die Positionsdaten der Roboter und stellt diese zur Verfügung.

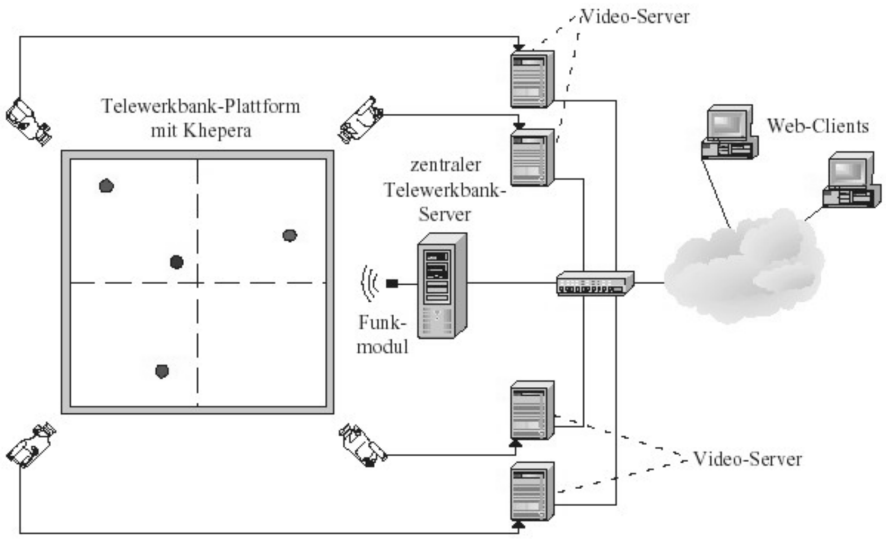

Abbildung 6-2. Aufbau der Telewerkbank. Kameras beobachten die Operationsfläche. Die Video-Server werten die Bilddaten aus und stellen den Robotern ihre Positionsdaten drahtlos in Echtzeit zur Verfügung. Die Steuerung der Experimente kann über das lokale Netzwerk oder auch über das Internet erfolgen.

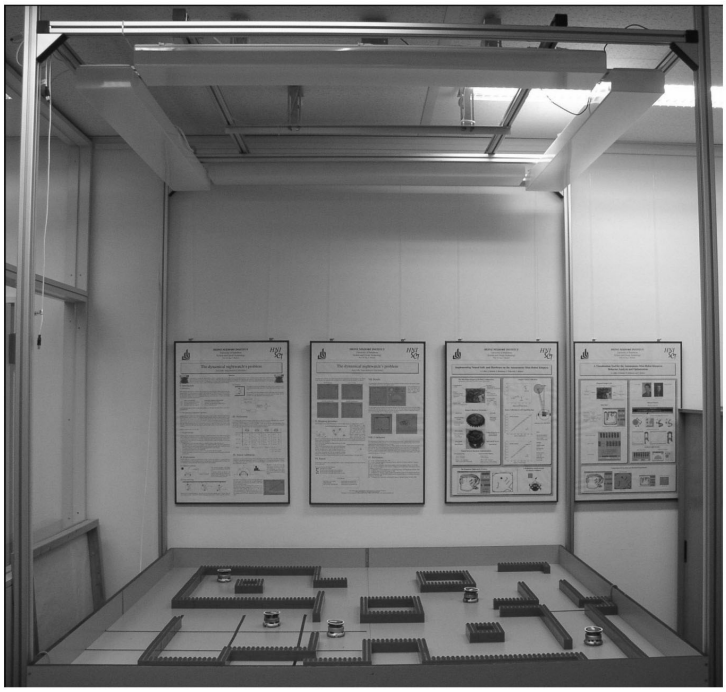

Abbildung 6-3. Ein Foto der Operationsfläche der Telewerkbank.

6.2 Implementierung des CSD-Protokolls

Eine vereinfachte Version des Protokolls wurde für ein mit Bluetooth-Kommunikationsmodulen bestücktes Multi-Roboter-System aus Khepera-Robotern implementiert. Eine Anpassung des Protokolls ist sinnvoll, weil durch Bluetooth bereits eine Netzwerktopologie in Form von miteinander verbundenen Piconetzen vorgegeben wird. Ein solcher Verbund aus Piconetzen, auch als Scatternetz bezeichnet, wurde in [Stru06] mit den gleichen wie auch hier verwendeten Bluetooth-Kommunikationsmodulen realisiert. Die Struktur des Scatternetzes kann hier direkt genutzt werden, anstatt eine weitere überlagerte Organisationsstruktur zu schaffen. Ein weiterer Grund ist, dass bei Roboterexperimenten nicht immer Positionsdaten zur Verfügung stehen. Die Dienst-Entdeckung sollte aber auch dann funktionieren. Wenn Positionsdaten zur Verfügung stehen sollten, werden sie selbstverständlich bei der Auswahl von Dienstleistern berücksichtigt. Die Wahl der Hauptknoten muss allerdings auch ohne Positionsdaten erfolgen können. Hier ist in diesem Fall passenderweise der erstgenannte Beweggrund für die Anpassung von Nutzen. Für die Bestimmung der Hauptknoten ist es möglich, die durch Bluetooth vorgegebene Piconetz-Struktur direkt

zu verwenden. Bluetooth-Piconetze sind in Sterntopologie aufgebaut mit dem Master des Piconetzes im Zentrum. Diese Master können als Hauptknoten für die Dienst-Entdeckung verwendet werden. Die Knoten eines Piconetzes informieren ihren Master regelmäßig über die von ihnen angebotenen Dienste. Wenn ein Knoten über eine Funktionalität nicht selbst verfügt und einen Dienst sucht, wird eine Suchanfrage an den Master gesendet. Wenn der Dienst nicht im eigenen Piconetz verfügbar ist, wird die Suchanfrage (im Scatternetz) von Master zu Master weitergeleitet, bis der gewünschte Dienst gefunden oder eine festgelegte Suchweite überschritten ist.
Die anderen Code-Teile der Simulation und der Implementierung im reellen Multi-Roboter-System stimmen überein.

Die Dienst-Entdeckung wird in drei Prozessen umgesetzt. Ein Prozess ist dabei verantwortlich für den Empfang und die Verarbeitung eintreffender Nachrichten. Der zweite Prozess informiert den Master eines Knotens regelmäßig über Änderungen im Dienstangebot und die aktuelle Position des Knotens, falls Positionsdaten zur Verfügung stehen. Für die Nutzung der Dienst-Entdeckung reichen wenige Befehle aus, die im Benutzerprozess aufgerufen werden. Der Nachrichtenaustausch und Suchvorgänge erfolgen völlig transparent für den Nutzer der Dienst-Entdeckung. Für Dienstleister sind dabei insbesondere die Befehle *addService* und *deleteService* von Bedeutung, um neue Dienste registrieren und bestehende Dienste wieder abzumelden zu können. Potentielle Kunden nutzen wiederum den Befehl *searchService*, um Dienste zu lokalisieren (Abb. 6-4). Die Integration der Dienst-Entdeckung in andere Programme beschränkt sich weitgehend auf die Einbindung der neuen Programmteile beim Kompiliervorgang, der Aktivierung der Schnittstelle für die Weitergabe der nicht für die Dienst-Entdeckung bestimmten Nachrichten und schließlich der Aktivierung der Schnittstelle für das Auslesen von Positionsdaten. Ein ausführliches Beispiel für die Integration und Nutzung der Dienst-Entdeckung findet sich in A.9.

```
...
// service provider: register service
addService(serviceDescription);
sendUpdate();   //this is optional. Either send update to master
        //immediately or let the update process take care of that...
```

```
// service provider: de-register service
deleteService(serviceDescription);
sendUpdate();   //this is optional. Either send update to master
        //immediately or let the update process take care of that...

// customer: discover service
searchService(serviceDescription);   //this is a non-blocking call
        //Either wait for its completion (like in this example) or
        //continue processing and check back later
while (!serviceSearchCompleted())   //wait until service search is
        //completed
   sleep(1);
if (getServiceProviderID() != -1)   //if successfully located a
        //suitable service provider, print id of service provider
   printf("Found service provider %d.\n", getServiceProviderID());
...
```

Abbildung 6-4. Code-Beispiele für das Hinzufügen, Entfernen und Suchen von Diensten.

6.3 Beispielszenario

Es wurde ein Beispielszenario für die Anwendung des Dienst-Entdeckungs-Protokolls und zur Demonstration einiger Vorteile des servicebasierten Ansatzes zur Koordinierung von Multi-Roboter-Systemen entwickelt (Abb. 6-5 bis 6-13). Hierfür werden Khepera-Roboter ausgestattet mit Bluetooth-Kommunikationsmodulen und die Telewerkbank verwendet, die den Robotern drahtlos ihre globalen Positionen in Echtzeit zur Verfügung stellen kann.

Drei Typen von Robotern sind im Einsatz: Roboter ausgestattet mit Kameramodulen, Transportroboter ausgestattet mit Greifern, und Patrouillenroboter, die außer dem Kommunikationsmodul keine zusätzlichen Module mehr tragen.

Einem Kameraroboter wird die Aufgabe zugewiesen, alle Zylinder innerhalb des Operationsgebiets an einer Zielposition zu finden und zusammenzutragen. Er fährt zufällig herum und sucht nach Zylindern. Nach der Entdeckung eines Zylinders lokalisiert der Kameraroboter einen Transportroboter und ruft ihn zu Hilfe, da der Kameraroboter selbst keine Möglichkeit hat, den Zylinder zu transportieren. Die Position des Zylinders wird durch Ausrichten des Roboters auf den Zylinder und

anschließender Messung der Breite des Zylinders auf dem Kamerabild zur Abschätzung der Entfernung bestimmt. Anschließend werden die Positionskoordinaten des Zylinders an den Transportroboter übermittelt. In der oberen rechten Ecke des Operationsgebiets patrouilliert ein Roboter in einem abgeschlossenen Gebiet. Der Patrouillenroboter repräsentiert einen preiswerten, schnellen und einfachen Roboter. Bei Entdeckung eines potentiellen bedeutsamen Ereignisses, zum Beispiel einer Lücke in der Wand, ruft er einen fortgeschritteneren Roboter zu Hilfe, um die Situation zu bewerten. Der herbeigerufene Roboter, in diesem Fall der Kameraroboter, könnte beispielsweise mit komplexeren und teureren Sensoren oder Werkzeugen zur Bewertung oder Reparatur des Schadens ausgestattet sein. Der Kameraroboter dient nun zwei verschiedenen Zwecken. Im ersten Fall nutzt er einen Dienst, während er im zweiten Fall einen Dienst zur Verfügung stellt. Da sowohl das Zusammentragen der Zylinder als auch das Patrouillieren einen Kameraroboter benötigt, wird ein zweiter Kameraroboter hinzugefügt. Der zweite Kameraroboter wird automatisch und nahtlos in das Gesamtsystem eingefügt. Es ist für ihn nicht notwendig, alle anderen Roboter sofort über seine Existenz zu informieren. Wenn seine Dienste benötigt werden, erfahren die anderem Roboter im Verlauf der Dienst-Entdeckung von seiner Verfügbarkeit. Die Dienste, die ausgetauscht werden können, sind nicht auf physischen Interaktionen oder Interaktionen zwischen Robotern beschränkt. Im Rahmen dieses Beispielszenarios wird zum Schluss ein menschlicher Benutzer eingeschaltet, der Infrarot-Sensordaten von der unteren Begrenzung des Operationsgebiets haben möchte. Der geografisch nächste Roboter, der in der Lage ist, diese Daten bereitzustellen, wird aus dem Operationsgebiet rekrutiert. Weder die Kameraroboter noch die Greifer haben normalerweise die Aufgabe, Infrarot-Sensordaten an andere Roboter oder menschliche Benutzer zu liefern. Aber einer von ihnen ist einfach zu dem Zeitpunkt und an dem Ort der geeignete Roboter, um diese Aufgabe durchzuführen. Hier sieht man im Übrigen auch, dass jeder Roboter eine Vielzahl von Diensten anbieten kann.

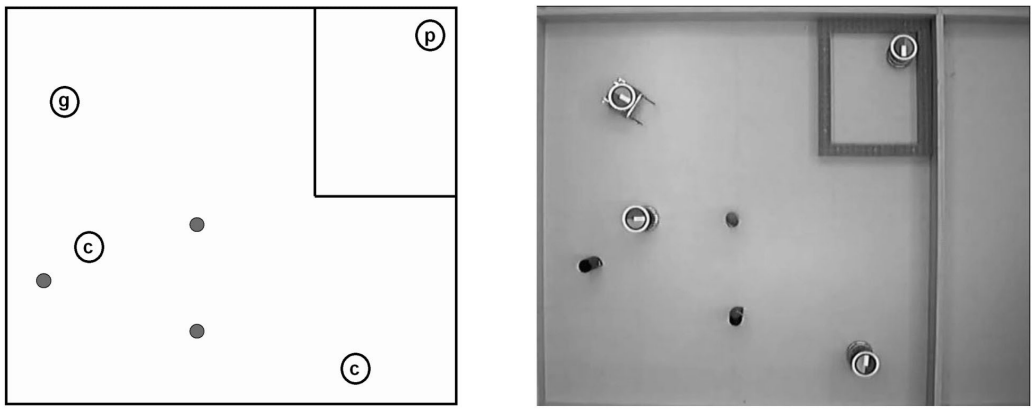

Abbildung 6-5. Der Aufbau des Szenarios. Es gibt einen Greifroboter (g), zwei Kameraroboter (c), von denen der untere rechte anfangs deaktiviert ist, und einen Patrouillenroboter (p).

Abbildung 6-6. Nach der Entdeckung des ersten Zylinders lokalisiert der Kameraroboter den Greifroboter und nutzt dessen Transportdienst, da er selbst keine Zylinder transportieren kann.

Abbildung 6-7. Der Kameraroboter sucht weiter nach Zylindern. Von außen wird ein Schaden im Mauerwerk nachgestellt, der vom Patrouillenroboter entdeckt wird.

Abbildung 6-8. Der Patrouillenroboter ruft seinerseits nun den Kameraroboter zu Hilfe, um die Situation zu bewerten. Diesmal ist der Kameraroboter der Dienstleister. Danach sucht der Kameraroboter weiter nach Zylindern. Das Loch in der Wand wird von außen geschlossen.

Abbildung 6-9. Der Patrouillenroboter fährt wieder. Nun wird der zweite Kameraroboter unten rechts aktiviert.

Abbildung 6-10. Der zweite Zylinder wurde gefunden und wieder wird der Greifroboter gerufen.

Abbildung 6-11. Der Greifer transportiert den zweiten Zylinder ab. Es wird erneut ein Loch im Mauerwerk nachgestellt.

Abbildung 6-12. Diesmal wird jedoch der zweite Kameraroboter gerufen, um den Schaden im Mauerwerk zu untersuchen, während der erste Kameraroboter weiter nach Zylindern sucht. Der zweite Kameraroboter wurde automatisch und nahtlos in das Gesamtsystem integriert.

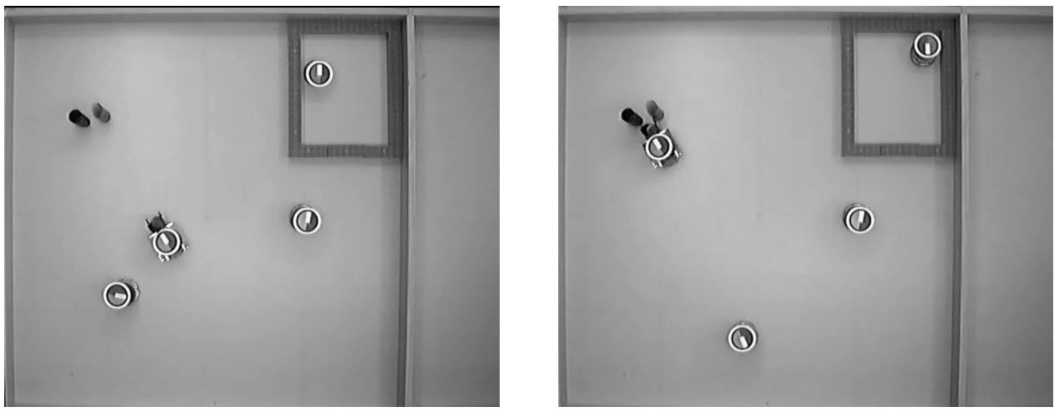

Abbildung 6-13. Der letzte Zylinder wird abtransportiert.

6.4 Integration einer Web-Service-Schnittstelle in den Khepera

In diesem Abschnitt wird beschrieben, wie eine Web-Service-Schnittstelle in den Khepera-Roboter integriert wird. Wie in 7.2 diskutiert wird, ist die Dienst-Entdeckung nur eine – wenn auch wichtige – Komponente bei der Entwicklung servicebasierter Multi-Roboter-Systeme. Vorteilhafterweise stimmen viele Fragen und Herausforderungen bei der Entwicklung servicebasierter Roboter-Systeme mit den Problemstellungen in verwandten Gebieten überein. Für Multi-Roboter-Systeme ist dabei die Forschung im Bereich der Softwareagenten und allgemein die Forschungsarbeit über die Kooperation heterogener, interoperabler Programme in großen Netzwerken von Bedeutung. Auch hier sind Fragestellungen wie die sinnvolle Zerlegung komplexer Aufgaben und das Zusammenfügen verschiedener Programme bzw. Dienste für die Bearbeitung dieser Aufgaben zu lösen. Insofern erscheint es sinnvoll, bei der Entwicklung servicebasierter Multi-Roboter-Systeme bereits bestehende Lösungen und Erfahrungen zu nutzen.

Durch die Verwendung von Web-Services, die für die Entwicklung von serviceorientierten Anwendungen im Internet konzipiert wurden, können viele Herausforderungen bei der Entwicklung servicebasierter Multi-Roboter-Systeme gelöst werden. Man wird sehen, dass für die Integration einer Web-Service-Schnittstelle in einen Roboter nur Anpassungen auf der untersten Ebene der Web-Service-Architektur, der Transportschicht, und der obersten Ebene der Web-Service-Architektur, der Discovery-Schicht, notwendig sind. Das in dieser Arbeit entwickelte CSD - Protokoll

kann dazu verwendet werden, um die Discovery-Schicht durch eine für Roboternetzwerke geeignete Lösung zu ersetzen.
Es sei angemerkt, dass Web-Services nur eine mögliche Ausprägung einer serviceorientierten Architektur darstellen und exemplarisch für die Implementierung herausgegriffen wurden.

A. Web-Services

Web-Services sind ein Standard, der vom W3C Konsortium für die Entwicklung serviceorientierter Anwendungen im Internet konzipiert wurde. Ein Web-Service ist eine Anwendung, die es seinen Nutzern erlaubt, über das Netzwerk und durch offene, standardisierte Schnittstellen auf seine Funktionalität zuzugreifen. Ein großer Vorteil von Web-Services ist deren Interoperabilität. Die Funktionalität eines Web-Services kann unabhängig von Ort, Plattform und Programmiersprache genutzt werden. Web-Services stellen ihre Funktionalität gekapselt zur Verfügung, so dass sie genutzt und wieder verwendet werden können, ohne Implementierungsdetails zu kennen. Die lose Kopplung von Web-Services bietet die Möglichkeit, verschiedene Dienste zu kombinieren, um komplexere, integrierte Dienste anzubieten.

Für die Nutzung eines Web-Services sucht man im ersten Schritt den gewünschten Dienst in einem bekannten, zentralen Verzeichnis, z. B. einen Dienst für die Wettervorhersage. Man erhält nähere Angaben in Form einer so genannten *WSDL*-Datei (*Web Service Description Language*), die den Dienst beschreibt. Sie enthält außerdem Informationen über den Dienstanbieter und darüber, wie man auf den Dienst zugreift. Diese WSDL-Datei kann dazu verwendet werden, um automatisiert Zugriffsfunktionen für diesen Dienst zu generieren, um diese in die eigene Anwendung zu integrieren. Dieses Beispiel illustriert die Eigenschaft der Interoperabilität und die Möglichkeit zur automatisierten maschinellen Verarbeitung. Es ist nicht von Bedeutung, was sich auf der Seite des Dienstanbieters abspielt. Solange sich der Dienstnutzer an die vorgegebenen Standards hält, kann der Zugriff vollautomatisiert werden.
Ein weiteres Beispiel ist die Kombination zweier Dienste. Der eine Dienst stelle Zitate bereit, der andere übersetze Texte zwischen verschiedenen Sprachen. Eine Kombination dieser beiden ergibt einen Dienst, der Zitate mehrsprachig zur Verfügung stellen kann. Dies ist ein Beispiel für das Zusammenfügen von Diensten, um einen Mehrwert zu

erzeugen. Dienste verschiedener Unternehmen können unabhängig von Plattform und Implementierungsdetails kombiniert werden, um komplexere Dienste zu erzeugen. Genau die Art von Kooperation, die auch für Multi-Roboter-Systeme angestrebt wird.
Dieselben Vorteile könnten in der Robotik sowohl für Interaktionen zwischen Robotern, aber auch für Interaktionen zwischen Robotern und menschlichen Benutzern oder Softwareagenten genutzt werden. Allerdings stellt die Dienstsuche über ein bekanntes, zentrales Verzeichnis, wie sie bei Web-Services erfolgt, eine Schwachstelle bei der Anwendung von Web-Services in dynamischen, drahtlosen Netzwerken ohne Infrastruktur dar.

Die Web-Service-Architektur besteht aus fünf Schichten [Enge04]. Die Transportschicht bildet die unterste Schicht der Architektur. Oft wird hier das weit verbreitete *Hypertext Transfer Protocol* (HTTP) für die Kommunikation verwendet. Die Informationsschicht enthält XML-formatierte (*eXtended Markup Language*) Nachrichten. Das Kodieren von Daten im XML-Format wird auch als XML-Serialisierung bezeichnet. Ausgehende Nachrichten werden serialisiert und eintreffende Nachrichten werden für die weitere Verarbeitung dekodiert. Die Verpackungsschicht verwendet das XML-basierte SOAP Protokoll (Simple Object Access Protocol). SOAP ist ein plattform- und sprachunabhängiges, nachrichten-basiertes Protokoll für die Kommunikation zwischen Anwendungen. Die Service-Schicht verwendet die *Web Service Description Language* (WSDL), um einen Web-Service und die von ihm zur Verfügung gestellte Funktionalität zu beschreiben. Schließlich gibt es noch die Discovery-Schicht, die Mechanismen für die Bekanntmachung und Entdeckung von Web-Services zur Verfügung stellt, üblicherweise basierend auf der UDDI-Spezifikation (Universal Description, Discovery, and Integration) (siehe 2.2.1.3).

B. Implementierung

Der Khepera II Roboter wird als Basisplattform für die Implementierung gewählt, auch um die niedrigen Hardwareanforderungen für die Integration solch einer serviceorientierten Architektur zu demonstrieren. Als Ergebnis wird der Zugriff auf den Roboter über eine Web-Service-Schnittstelle ermöglicht (Abb. 6-16).
Für die Kommunikation der Roboter wird ein Bluetooth-Modul verwendet. Die Telewerkbank wird eingesetzt, um dem Benutzer des Roboters ein Video des

Operationsgebiets in Echtzeit zur Verfügung zu stellen. Für die Umsetzung des Web-Services auf dem Khepera wurde gSOAP verwendet. gSOAP ist eine Open Source Entwicklungswerkzeug für Web-Services [Enge04]. Es unterstützt reinen C Code und generiert Code von geringem Umfang. Zudem kann die Transportschicht flexibel angepasst und durch roboterspezifische Transportschichten ersetzt werden. Es wird außerdem die Kompression und das Streaming von XML-Daten unterstützt, so dass das Nachrichtenaufkommen reduziert und eine sofortige Verarbeitung möglich wird, was besonders in drahtlosen Roboternetzwerken von Bedeutung ist. Schließlich ist der Quellcode frei verfügbar, so dass er an die hier verwendete Roboterplattform angepasst werden kann.

Die Entwicklung von Web-Services mit gSOAP funktioniert wie folgt. Aus einer Spezifikation der über das Netzwerk anzubietenden Funktionen und der verwendeten Datentypen, beispielsweise in Form einer C Kopf-Datei, wird ein Codegerüst für Funktionsaufrufe und für die Serialisierung der verwendeten Datentypen erzeugt. Dieser Code wird zusammen mit gSOAP Bibliotheksfunktionen und der eigentlichen Anwendung verbunden, um die Funktionalität der Anwendung als Web-Service anzubieten. Weiterhin wird eine Dienstbeschreibung in Form einer WSDL-Datei generiert, die von Kunden genutzt werden kann, um automatisiert Zugriffsfunktionen für diesen Dienst zu erzeugen (Abb. 6-14).

```
...
<!-- operation request element -->
<element name="Goto">
 <complexType>
  <sequence>
   <element name="i-X" type="xsd:int" minOccurs="1" maxOccurs="1"/>
   <element name="i-Y" type="xsd:int" minOccurs="1" maxOccurs="1"/>
   <element      name="i-Alpha"      type="xsd:int"      minOccurs="1"
maxOccurs="1"/>
  </sequence>
 </complexType>
</element>
<!-- operation response element -->
<element name="GotoResponse">
 <complexType>
  <sequence>
   <element      name="pi-Result"      type="xsd:int"      minOccurs="0"
```

```
maxOccurs="1" nillable="true"/>
  </sequence>
 </complexType>
</element>
...
```

Abbildung 6-14. Ein Beispiel für die Darstellung einer Funktion in WSDL. Hier ist die *Goto*-Funktion dargestellt, an die x, y und Alpha für die anzufahrende Position und Orientierung übergeben werden. Die Beschreibung der Funktionen im XML-basierten WSDL erlaubt eine automatisierte maschinelle Generierung von Zugriffsfunktionen zur Nutzung des Dienstes.

Web-Services sind aufgrund der verwendeten Standards nicht geeignet für eine harte Echtzeitsteuerung von Robotern. Die verwendeten Protokolle sind nicht echtzeitfähig und die unbestimmte Latenz von drahtlosen Kommunikationsverbindungen verhindert eine ständige direkte Kontrolle des Roboters. Daher erfordert die Fernsteuerung durch Web-Services einen gewissen Grad an Autonomie von den Robotern. Autonomie ist ebenfalls notwendig, da oft nur die Roboter selbst ihre Umwelt und aktuelle Situation vollständig erfassen und die Gültigkeit und Sicherheit übermittelter Kommandos verifizieren können. Folglich wurden für die Bedienung des Roboters eine Anzahl grundlegender, modularer Kommandos zur Kontrolle des Roboters zur Verfügung gestellt. Beispielsweise wird eine *Goto*-Funktion bereitgestellt, an die die Parameter x, y und Alpha für Position und Orientierung übergeben werden.

In Abbildung 6-15 ist die Systemarchitektur der Implementierung dargestellt. Ein Nutzer im Netzwerk generiert sich aus der WSDL-Datei Zugriffsfunktionen, mit denen er auf den Roboter zugreifen kann. Die Befehle an den Roboter werden von einem Server, der an das Netzwerk angeschlossen ist, drahtlos an den Roboter weitervermittelt. Für eine Nutzung in einem Roboternetzwerk wäre kein Server notwendig, sondern es würde ein direkter Austausch zwischen den beteiligten Robotern erfolgen. Der Server dient hier nur für die Anbindung des Roboters an das Internet. Der Roboter verarbeitet die Anweisungen und führt die Befehle aus. Kameras über der Operationsfläche nehmen Bilder auf, die über einen Server live zur Verfügung stehen und dem Benutzer eine visuelle Rückmeldung seiner an den Khepera gesendeten Befehle geben. Die hier vorliegende Implementierung belegt dabei etwa 300 KiB im Arbeitsspeicher des Kheperas.

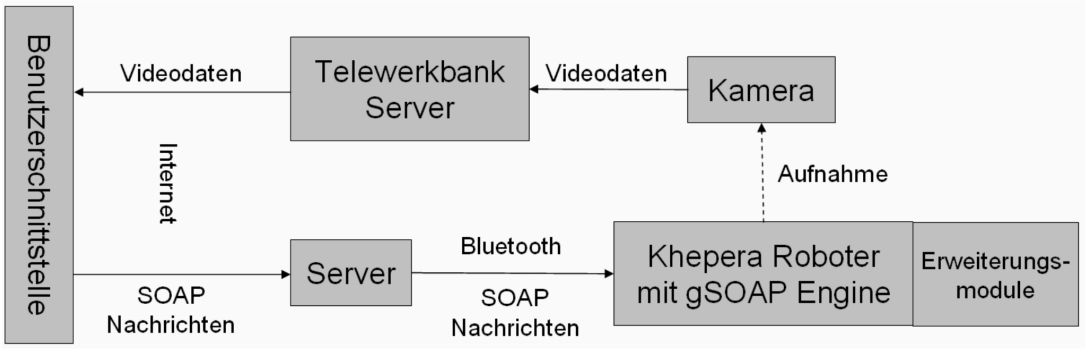

Abbildung 6-15. Illustration der Systemarchitektur. Der Roboter ist ein Web-Service. Befehle aus dem lokalen Netzwerk oder dem Internet werden über eine Bluetooth-Verbindung an den Roboter gesendet. Über das Kamerasystem der Telewerkbank erhält der Benutzer eine visuelle Rückmeldung.

Abbildung 6-16. Ein Foto der allein aus der Dienstbeschreibung des Roboters erstellten (in diesem Beispiel Web-Browser-basierten) Benutzeroberfläche. Hiermit kann der Roboter angewiesen werden, beliebige durch den Benutzer bestimmte Positionen auf dem Operationstisch anzufahren.

C. Notwendige Anpassungen für die Robotik

Aus den fünf Schichten der Web-Service-Architektur müssen für eine Anwendung in der Robotik insbesondere zwei Schichten genauer betrachtet werden. Einmal auf der untersten Ebene die Transportschicht, die gegebenenfalls durch roboterspezifische Transportschichten ersetzt werden muss. Und auf der obersten Ebene die Discovery-Schicht, weil das Entdecken von Diensten basierend auf UDDI nicht für mobile drahtlose Netzwerke ohne Infrastruktur geeignet ist (siehe dazu auch 2.2.1.3). Vom W3C wird zwar zusätzlich auch ein Peer-to-Peer-Ansatz für die Dienst-Entdeckung vorgeschlagen, dieser entspricht allerdings dem klassischen Fluten des Netzwerks und ist für größere Netzwerke sehr nachrichtenaufwendig. Dabei wird eine Suchanfrage im Netzwerk verbreitet, bis ein geeigneter Dienstanbieter gefunden oder ein Endkriterium erfüllt ist, zum Beispiel eine maximale Anzahl von Weiterleitungen. Bei der Integration von Web-Service-Schnittstellen in mobile Roboter fehlen somit geeigneten Mechanismen zur Dienst-Entdeckung, um Ergebnisse aus diesem Gebiet der serviceorientierten Architekturen auch für Multi-Roboter-Systeme nutzbar zu machen. Genau für diese Problematik bietet CSD eine Lösung.

6.5 Zusammenfassung

In diesem Kapitel wurde beschrieben, wie eine vereinfachte Version von CSD auf einem realen, mit Bluetooth-Kommunikationsmodulen bestückten Multi-Roboter-System umgesetzt wurde. Die Nutzung der Dienst-Entdeckung ist einfach und kommt mit wenigen Befehlen aus. Die Dienst-Entdeckung selbst erfolgt transparent für den Benutzer. Die Lösung kann in bestehende Programme integriert werden und erfordert im Wesentlichen nur die Einbindung der neuen Programmteile bei der Kompilierung und der Aufruf einiger vordefinierter Schnittstellen im Programmcode. Die Implementierung wird in einem Beispielszenario eingesetzt und einige Vorteile servicebasierter Multi-Roboter-Systeme werden demonstriert, wie z. B. die Möglichkeit zur Anforderung von Unterstützung aus der Umgebung, die dynamische Kooperation von Robotern und die nahtlose Integration neuer Roboter in das Gesamtsystem.

Im letzten Abschnitt wird schließlich eine Web-Service-Schnittstelle in einen mobilen Roboter integriert. Damit wird exemplarisch gezeigt, dass aktuelle Entwicklungen im Bereich der serviceorientierten Softwarearchitekturen teilweise direkt in servicebasierten Multi-Roboter-Systemen genutzt werden können und dies nur mit relativ geringen Hardwareanforderungen verbunden ist. Für die Integration einer Web-Service-Schnittstelle in einen mobilen Roboter sind nur Anpassungen auf der untersten Ebene der Web-Service-Architektur, der Transportschicht, und der obersten Ebene, der Entdeckungsschicht, erforderlich. Genau für den zweiten Punkt bietet CSD eine Lösung.

7. Zusammenfassung

Erste Ansätze in Richtung servicebasierter Multi-Roboter-Systeme lassen sich beispielsweise in [Park98] aus dem Jahre 1998 finden. Die Identifikation der Dienst-Entdeckung als eine essentielle Komponente und gleichzeitig als noch ungelöstes Problem für eine Umsetzung dieses Ansatzes bildet die Grundlage der vorliegenden Arbeit.

Dienst-Entdeckung erlaubt es Robotern, Dienste in großen, offenen, heterogenen Netzwerken zu entdecken und auszutauschen. Solch ein Austausch generischer Dienste zwischen spezialisierten, lose gekoppelten Robotern ist ein möglicher Ansatz für die Entwicklung großer Multi-Roboter-Systeme. Roboter können dann bei Bedarf andere Roboter für bestimmte Aufgaben aus der Umgebung rekrutieren. Durch die Verwendung offener, standardisierter Schnittstellen wird Interoperabilität sichergestellt, so dass die Funktionalität eines Roboters von jedem anderen Roboter genutzt werden kann und verschiedene Robotertypen und Plattformen unterschiedlicher Hersteller miteinander interagieren können. Verschiedene Roboter können kooperieren und flexibel kombiniert werden, um neue, höherwertigere Dienstleistungen anzubieten. Es ist möglich, die Implementierung einzelner Roboter zu ändern, ohne gleich Auswirkungen auf das Gesamtsystem bedenken zu müssen, und Roboter können einfach ersetzt, hinzugefügt und entfernt werden. Servicebasierte Architekturen haben somit das Potential, die Entwicklung und Nutzung großer Multi-Roboter-Systeme stark zu vereinfachen. Eine effiziente Dienst-Entdeckung ist eine essentielle Komponente für solch eine servicebasierte Multi-Roboter-Architektur und bildet einen wichtigen Bestandteil für einen erfolgreichen Einsatz mobiler Roboter in unserer Umwelt.

In dieser Arbeit wurde das Cell-based Service Discovery-Protokoll (CSD) für die die Dienst-Entdeckung in Roboternetzwerken entwickelt[5]. Die Grundidee von CSD basiert auf einer Gitterstruktur aus Zellen mit speziellen Hauptknoten in jeder Zelle. Durch Einführung dieser Zellstruktur minimiert CSD im Unterschied zu anderen Protokollen die Auswirkungen von Knotenbewegungen auf das Gesamtnetzwerk. Die Hauptknoten

[5] Selbstverständlich eignet sich die Lösung dessen ungeachtet für alle Arten von mobilen Ad-hoc-Netzwerken, die durch die gleichen Eigenschaften gekennzeichnet sind, insbesondere eine hohe Mobilität der Teilnehmer, die Verfügbarkeit von Positionsinformationen in den Knoten und ein hohes Verhältnis von Dienstbekannt-machungen zu Dienstsuchen.

verwalten die Dienste der gewöhnlichen Knoten in ihren Zellen und Suchanfragen werden an sie gerichtet. Wenn ein gesuchter Dienst in der eigenen Zelle nicht verfügbar ist, werden Suchanfragen an Hauptknoten der Nachbarzellen gesendet. Durch verschiedene Erweiterungen wie das Suchen in mehreren Suchschritten kann der Nachrichtenaufwand weiter reduziert werden. Zusätzlich unterstützt CSD dynamische Erweiterungen und Verschiebungen des Operationsgebietes, semantisch starke Dienstbeschreibungen und eine nahtlose Integration bestehender Infrastruktur.

CSD wird analytisch und in Simulation untersucht und mit anderen Lösungen verglichen. CSD entfaltet seine Stärken in Szenarien, in denen eine skalierbare Lösung für Netzwerke mit mittlerer bis hoher Mobilität der Knoten benötigt wird. Durch die Organisation in Zellen werden die Kosten für Bekanntmachungen niedrig gehalten und die Auswirkungen von Knotenbewegungen verringert und damit auch die Robustheit der Lösung erhöht. Wenn Dienste mehrfach im Netzwerk vorkommen, hat dies besondere Vorteile für CSD. Durch Repliken werden die Suchkosten von CSD erheblich reduziert, die Kosten von Bekanntmachungen sind ohnehin schon niedrig. Die Existenz mehrerer gleichwertiger Dienste im Netzwerk wird ausgenutzt, um Entdeckungen effizient durchzuführen, die Robustheit des Systems beim Ausfall einzelner Dienste zu erhöhen und geografische Nähe von Kunde und Dienstleister sicherzustellen.

Abschließend wurde eine vereinfachte Version des Protokolls auf einem realen Multi-Roboter-System implementiert und anhand eines Beispielszenarios wurden die Vorteile servicebasierter Multi-Roboter-Systeme demonstriert.

Zusammenfassend sind die Hauptbeiträge dieser Arbeit:

- Die Identifikation der Dienst-Entdeckung als Grundproblem für dienst-basierte Multi-Roboter-Systeme.
- Die Entwicklung des Cell-based Service Discovery Protokolls (CSD) für die Dienst-Entdeckung in Roboternetzwerken unter besonderer Berücksichtigung der Voraussetzungen und Anforderungen von Robotersystemen.
- Die Analyse und der Vergleich des Protokolls mit anderen Lösungen für die Dienst-Entdeckung in mobilen Ad-hoc-Netzwerken.
- Die detaillierte Untersuchung des Protokolls in der Simulation.

- Die Implementierung einer vereinfachten Version des Protokolls auf einem realen Multi-Roboter-System und die Demonstration einiger Vorteile von Dienst-basierten Multi-Roboter-Systemen.

7.1 Weiterführende Arbeiten bezüglich CSD

Hier werden Hinweise für mögliche weiterführende Arbeiten bezüglich des CSD-Protokolls gegeben. Denkbar sind:

- Die Untersuchung des in 4.2.2.2 beschriebenen aktiven Austauschs zwischen Zellen für größere Netzwerke in der Simulation.
- Der Vergleich von CSD mit anderen Verfahren in der Simulation. Hierfür sind eigene Implementierungen der Verfahren notwendig, da diese bisher entweder nur theoretisch untersucht wurden (LANES) oder die Implementierungen nicht zur Verfügung stehen (RR und GCLP).
- Die Simulation in einem geeigneten Netzwerksimulator unter Berücksichtigung von Hindernissen, die sowohl Auswirkungen auf die Sendereichweiten wie auch auf das Bewegungsmodell der Knoten haben werden.
- Schließlich in Zukunft der Test und die Anwendung des Protokolls in einem großen, realen Netzwerk.

7.2 Weitere Herausforderungen in servicebasierten Multi-Roboter-Systemen

In dieser Arbeit liegt die Konzentration auf der Entdeckung von Diensten in Roboternetzwerken, was die unentbehrliche Grundkomponente für eine servicebasierte Multi-Roboter-Architektur bildet. Es gibt jedoch eine Vielzahl von Herausforderungen, die zusätzlich gelöst werden müssen (Abb. 7-1). Unter anderem wird eine gemeinsame Ontologie benötigt, die es Robotern erlaubt zu kommunizieren, und Aufgaben und Dienste zu beschreiben [SM05, JZ00]. Weiterhin ist ein Anreizsystem erforderlich, um kooperatives Verhalten zu belohnen, und Roboter zu motivieren, auch wirklich Dienste anzubieten [WM06]. Ansätze für die Lösung von Zugriffskonflikten müssen formuliert

und integriert werden. Erforderlich sind auch Strategien für eine adäquate Zerlegung von komplexen Aufgaben in Teilaufgaben [OVM05, PT06] und deren Zuweisung an die einzelnen Roboter [GM04]. Im implementierten Beispielszenario wurden einfache Lösungen gewählt: Die Aufgabenbeschreibungen sind proprietär, alle Roboter erbringen Dienste für alle anderen Roboter, die Bearbeitung von Aufgaben erfolgt nach der Reihenfolge des Eintreffens der Anfragen und die Zerlegung der Aufgaben ist vorgegeben. Um jedoch die Vorteile von servicebasierten Multi-Roboter-Systemen wirklich nutzen zu können, werden ausgereifte Lösungen für diese Fragestellungen notwendig sein, an denen aber bereits seit einiger Zeit mit Erfolg geforscht wird.

Der Sicherheitsaspekt wurde in dieser Arbeit nicht betrachtet. Die speziellen Charakteristika mobiler Ad-hoc-Netzwerke wie funkbasierte Multi-Hop-Kommunikation, das Fehlen von Infrastruktur und die Dynamik der Netzwerktopologie erhöhen ihre Angreifbarkeit [RMCC+06]. Für die Dienst-Entdeckung sind neben Aspekten wie Vertraulichkeit und Integrität der versendeten Nachrichten eine hohe Verfügbarkeit der Dienste sowie die Authentifizierung und Autorisierung der Teilnehmer von besonderer Bedeutung. Einerseits geht es darum, Störungen des Betriebs durch feindselige Teilnehmer zu unterbinden, andererseits müssen Dienstnutzer und Dienstanbieter eindeutig identifiziert und es dürfen nur berechtigte Zugriffe auf Dienste zugelassen werden.

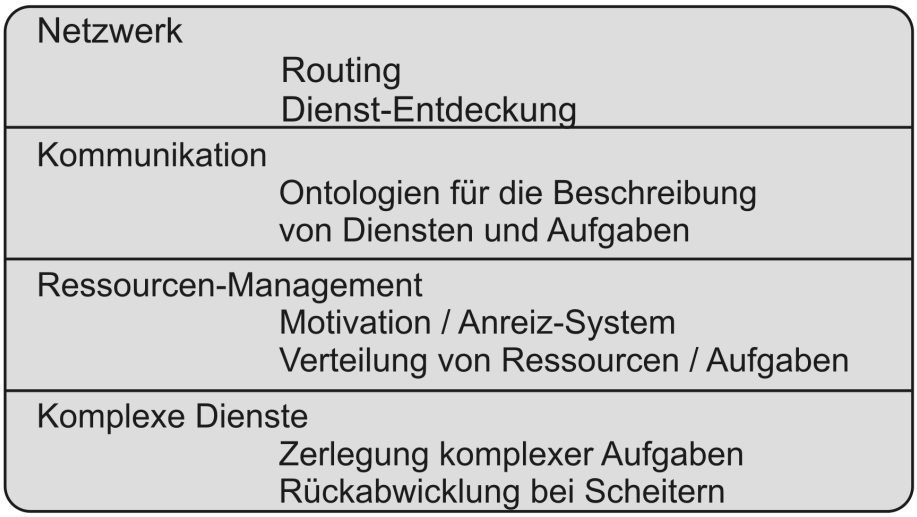

Abbildung 7-1. Zu lösende Fragestellungen im Forschungsgebiet Multi-Roboter-Systeme.

Anhang

A.1 Kommunikationsdistanzen

A. Mittlere Distanzen zweier zufällig gewählten Punkte

Die folgenden Integrale wurden mit Hilfe des Computer-Algebra-Systems *Mathematica* [Wolf] ausgewertet und die Ergebnisse durch Simulationenexperimente überprüft.

Die mittlere Manhattandistanz zweier zufällig gewählten Punkte $P_1(x_1,y_1)$ und $P_2(x_2,y_2)$ auf einer quadratischen Grundfläche der Seitenlänge a ($0 \le x_1,y_1,x_2,y_2 \le a$) beträgt

$$\int_{x_1=0}^{a}\int_{y_1=0}^{a}\frac{\int_{x_2=0}^{a}\int_{y_2=0}^{a}\frac{|x_2-x_1|+|y_2-y_1|}{a^2}dy_2dx_2}{a^2}dy_1dx_1=\frac{2}{3}a \qquad (a.1)$$

Werden zwei Punkte $P_1(x_1,y_1)$ und $P_2(x_2,y_2)$ zufällig auf einer rechteckigen Grundfläche mit den Seitenlängen a ($0 \le y_1,y_2 \le a$) und 2a ($0 \le x_1,x_2 \le 2a$) gewählt, beträgt die mittlere Manhattandistanz

$$\int_{x_1=0}^{2a}\int_{y_1=0}^{a}\frac{\int_{x_2=0}^{2a}\int_{y_2=0}^{a}\frac{|x_2-x_1|+|y_2-y_1|}{2a^2}dy_2dx_2}{2a^2}dy_1dx_1=a \qquad (a.2)$$

Entsprechend gilt für die euklidische Distanz zweier zufällig gewählten Punkte $P_1(x_1,y_1)$ und $P_2(x_2,y_2)$ auf einer quadratischen Grundfläche der Seitenlänge a ($0 \le x_1,y_1,x_2,y_2 \le a$)

$$\int_{x_1=0}^{a}\int_{y_1=0}^{a}\frac{\int_{x_2=0}^{a}\int_{y_2=0}^{a}\frac{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}{a^2}dy_2dx_2}{a^2}dy_1dx_1 \approx 0,52\cdot a \quad \text{(a.3)}$$

Für die mittlere Manhattandistanz von einer Ecke eines Quadrats der Seitenlänge a zu einem zufällig gewählten Punkt in diesem Quadrat gilt mit dem Eckpunkt $P_1(0,0)$ und einem zufällig gewählten Punkt $P_2(x,y)$ mit $0 \le x,y \le a$

$$\int_{x=0}^{a}\int_{y=0}^{a}\frac{x+y}{a^2}dydx = a \quad \text{(a.4)}$$

B. Mittlere Kommunikationsdistanz für Suchanfragen in LANES

Suchanfragen werden bei LANES horizontal in beide Richtungen verschickt. Wenn es keine Repliken im Netzwerk gibt, dann wird die Suchanfrage in der einen Richtung bis zu derjenigen Bahn weitergeleitet, die den gesuchten Dienst enthält und in der anderen Richtung bis zum Rand des Netzwerks. Wenn $0 \le x_1 \le a$ die Position des suchenden Knotens und $0 \le x_2 \le a$ die Position des gesuchten Dienstleisters ist, dann beträgt die Kommunikationsdistanz x_2 für $x_{2>}x_1$, 0 für $x_{2=}x_1$ und $a-x_2$ für $x_{2<}x_1$. Im Mittel ergibt sich für Suchanfragen in LANES damit eine Kommunikationsdistanz von

$$\int_{x_1=0}^{a}\int_{x_2=0}^{a}\frac{\begin{Bmatrix} x_2 & , x_2>x_1 \\ 0 & , x_2=x_1 \\ a-x_2 & , x_2<x_1 \end{Bmatrix}}{a^2}dx_2dx_1 = \frac{2}{3}a \quad \text{(a.5)}$$

A.2 Abschätzung der Nachrichtengrößen

Da es derzeit kaum Erfahrungswerte bezüglich der Größe von Dienstbeschreibungen in großen heterogenen Multi-Roboter-Systemen gibt, werden als Anhaltspunkt für die in der Simulation zu verwendenden Nachrichtengrößen Dienstbeschreibungen von Web-Services aus dem Internet herangezogen. So belegt beispielsweise die Beschreibung des Google Suchdienstes als Web-Service etwa 5 KiB, nach Komprimierung durch einen Standard-Komprimierungsprogramm unter Verwendung des Lempel-Ziv-Welch-Algorithmus etwa 1,5 KiB [Goog]. Die Beschreibung der Suchmaske des Buchhändlers Amazon belegt etwa 50 KiB in unkomprimierter und 3,5 KiB in komprimierter Form [Amaz]. Eine Reihe anderer untersuchter Web-Services hatten ebenfalls eine Größe zwischen 10 KiB bis 50 KiB in unkomprimierter und zwischen 1 KiB bis 5 KiB in komprimierter Form [XMET]. Diese Werte werden als Anhaltspunkt für die Größenordnung der in der Simulation verwendeten Dienstbeschreibungen genutzt. Dabei wird allerdings davon ausgegangen, dass für drahtlose Roboternetzwerke eine effizientere Codierungsform als XML (wie bei Web-Services) verwendet wird und durch eine gemeinsame Ontologie ebenfalls Ersparnisse möglich sind. Die Nachrichtengröße für eine Dienstbeschreibung bei einer Dienstsuche wird auf 512 Bytes gesetzt. Bei einer Bekanntmachung fallen für die Beschreibungen der Dienste 4096 Bytes an (8 Dienste pro Roboter).

A.3 Zufällige Generierung der Dienste auf den Knoten

Im Folgenden ist der C-Code dargestellt, der in der Simulation bei jedem Knoten für die (pseudo-) zufällige Generierung der eigenen Dienste verwendet wurde.

```
...
for (int i=0; i<numberOfServicesToAdd; i++)
{
  bool inserted = false;
  while (!inserted)
  {
    int k = int(Random::uniform(MAX_NUMBER_OF_DIFFERENT_SERVICES));
//0..MAX_NUMBER_OF_SERVICES-1
```

```
    int serviceNumber = int(Random::uniform(k+1)/1.1);
//0..MAX_NUMBER_OF_SERVICES-1
    searchOwnServices(serviceNumber);
    if (providerID == -1)
    {
      addService(serviceNumber);
      inserted = true;
    }
  }
}
...
```

Abbildung A-1. Code für die zufällige Generierung von Diensten auf den Knoten.

A.4 Abschätzung der Anfragerate

Prinzipiell müssen die Zeitabstände für Dienstsuchen mit den Zeiten für die Bearbeitung von Aufträgen korreliert sein. Andernfalls wäre das System langfristig nicht im Gleichgewicht. Zuerst wird hier daher die durchschnittliche Zeit für die Bearbeitung einer Aufgabe aus den Pausezeiten des Bewegungsmodells und den Weglängen abgeschätzt. Bei einer mittleren Geschwindigkeit der Roboter von 9 km/h wird für die Fahrt zwischen zwei zufällig gewählten Punkten im Operationsgebiet 60s (bei 300m x 300m) bis 80s (bei 400m x 400m) benötigt (siehe hierzu auch 5.3). Da mit Repliken im Netzwerk die Roboter Dienste in der Nähe in Anspruch nehmen werden, fließt dieser Wert jedoch nur zur Hälfte ein. Zusätzlich wird eine mittlere Bearbeitungszeit von 200 Sekunden vor Ort angenommen, was den Pausezeiten im Bewegungsmodell (5.3) entspricht. Damit ergibt sich eine mittlere Bearbeitungszeit pro Auftrag von

$$\bar{t}_B = \text{mittlere Pausezeit der Rob.} + \text{mittlere Bewegungszeit der Rob.} = 200s + 35s = 235s \quad \text{(a.6)}$$

Grundsätzlich kann argumentiert werden, dass die Pausen zwischen Suchanfragen nicht dauerhaft kürzer sein können als die mittlere Bearbeitungszeit pro Auftrag, damit das System langfristig im Gleichgewicht ist. Allerdings müssen hier noch weitere Einflussfaktoren berücksichtigt werden. Für eine höhere Anfragerate spricht, dass pro

Auftrag durchaus auch mehrere Dienstsuchen gestartet werden können, sei es, weil Konditionen der Dienstanbieter verglichen werden, weil ein gewünschter Dienst gar nicht im Netzwerk existiert oder Dienste kombiniert werden. Ebenso können Aufgaben teilweise auch parallel bearbeitet werden, beispielsweise der Transport von Objekten in ein Gebiet und die Bereitstellung von Sensordaten aus diesem Gebiet. Für eine niedrigere Anfragerate spricht wiederum, dass vermutlich nicht immer alle Roboter beschäftigt sind oder selbst keine Dienste in Anspruch nehmen. Es wird daher angenommen, dass pro tatsächlich erteiltem Bearbeitungsauftrag im Mittel vier Suchvorgänge durchgeführt werden, dass etwa ein Drittel der Aufgaben parallel bearbeitet werden können und dass im Mittel etwa ein Drittel der Roboter keine Aufträge versendet, weil sie für sich selbst keine Aufgaben zu erfüllen haben oder ihre Aufgaben allein bearbeiten können. Mit der zuvor hergeleiteten mittleren Bearbeitungszeit ergibt sich als Schätzung für die Anfragerate

$$r_A = 4 \cdot \frac{3}{2} \cdot \frac{2}{3} \cdot \frac{\text{Anfragen}}{\text{Bearbeitungszeit}} \approx 1 \cdot \frac{\text{Anfrage}}{60\text{s}} \qquad \text{(a.7)}$$

Diese Schätzung wird als grober Richtwert für die Anfragerate in der Simulation verwendet. In der Simulation wird die Anfragrate von 1 Anfrage/30s bis 1 Anfrage/90s variiert.

A.5 Abschätzung der möglichen Ersparnisse durch positionsbasiertes Multicasting

Betrachtet wird die obere linke Ecke eines Gebiets mit 16 Zellen. Da kein Multicasting verwendet wird, wird an jede zu durchsuchende Zelle eine separate Suchnachricht gesendet. Im ersten Suchschritt werden die ersten 8 Nachbarzellen durchsucht. Dies sind, wenn man die Zellen von links oben nach rechts unten zeilenweise aufsteigend nummeriert, die Zellen mit den Nummern 2, 3, 5, 6, 7, 9, 10 und 11. Für eine Suchanfrage an die ersten 8 Nachbarzellen sind damit ohne positionsbasiertes Multicasting 13 Nachrichtenübertragungen von Zelle zu Zelle erforderlich. Dies lässt sich durch einfaches Zählen bestimmen. Entsprechend sind für Suchanfragen an die restlichen 7 Zellen minimal 21 Nachrichtenübertragungen von Zelle zu Zelle

erforderlich, da für die Auslieferung der Suchnachrichten zuerst das bereits durchsuchte Gebiet durchquert werden muss. Es wird angenommen, dass 50 % der Anfragen bereits im ersten Suchschritt und 50 % erst im zweiten Suchschritt aufgelöst werden. Daraus ergeben sich für die mittleren Kosten der Suche ohne positionsbasiertes Multicasting

$$K_{-pbm} = 0,5 \cdot 13 + 0,5 \cdot 21 = 17 \qquad \text{(a.8)}$$

Nachrichtenübertragungen von Zelle zu Zelle.

Bei Anwendung von positionsbasiertem Multicasting wird für eine Suche nur eine einzelne Nachricht ausgesendet und erst bei Bedarf im oder kurz vor dem Zielgebiet kopiert und verteilt. Wiederum durch einfaches Zählen und den gleichen Randbedingungen wie oben ergeben sich für die mittleren Suchkosten unter Verwendung von positionsbasiertem Multicasting von

$$K_{+pbm} = 0,5 \cdot 8 + 0,5 \cdot 9 = 8,5 \qquad \text{(a.9)}$$

Nachrichtenübertragungen von Zelle zu Zelle.
Insgesamt ist so durch Anwendung von positionsbasiertem Multicasting schätzungsweise eine Verringerung der Suchkosten von bis zu 50 % in einem Operationsgebiet aus 16 Zellen möglich.

A.6 Abschätzung für die auf Fünfergruppen erweiterte Version von CSD

Hier wird abgeschätzt, ab welcher Netzwerkgröße ein Einsatz der erweiterten Version von CSD (4.2.2.2) sinnvoll ist. Die Notation ist wie in 4.3.

Zuerst wird die gewöhnliche Version von CSD ohne pro-aktiven Austausch zwischen Zellen betrachtet. Bei einem Zellwechsel sind eine Abmeldung aus der alten Zelle und eine Anmeldung in der neuen Zelle nötig. Die Kosten für Abmeldungen, bei denen lediglich eine Kontrollnachricht ohne weitere Daten verschickt wird, sind allerdings

gering im Vergleich zu den Kosten für Anmeldungen, bei denen Beschreibungen der Dienste verschickt werden, und werden daher vernachlässigt. Wie in Gl. (4.11) ergeben sich für die Kosten eines Zellwechsels

$$K_{1w} = k_b \cdot \sqrt{n_c} \tag{a.10}$$

Für eine Suche in einer Gruppe von 5 Zellen bei der gewöhnlichen Version von CSD muss fünfmal eine Suchnachricht von Zelle zu Zelle transportiert werden. Mit Gl. (4.12), das die Kosten des Versands einer Suchnachricht von einer Zelle in eine andere beschreibt, ergibt sich

$$K_{1s} = 5 \cdot k_s \cdot \sqrt{n_c} \tag{a.11}$$

Bei der erweiterten Version von CSD sind bei einem Zellwechsel 4 Abmeldungen für die vier ursprünglichen Nachbarzellen und 4 Anmeldungen für die vier neuen Nachbarzellen nötig. Auch hier werden die Kosten für Abmeldungen vernachlässigt. Für Dienstsuchen in der erweiterten Version sind etwa nur 2/5 der Kosten im Vergleich zur nicht erweiterten Version nötig. Es muss nur jede fünfte Zelle befragt werden, allerdings ist zusätzlich ein weiterer Zwischenschritt nötig, um Suchnachrichten vom Mittelpunkt einer Fünfergruppe zum Mittelpunkt der nächsten Fünfergruppe zu senden. Der Einfachheit halber wurde hier angenommen, dass der Austausch von Nachrichten zwischen diagonalen Zellen ebenfalls möglich ist und etwa dieselben Kosten verursacht wie bei einem horizontalen oder vertikalen Austausch. Als Kosten für Zellwechsel in der erweiterten Version von CSD ergeben sich damit

$$K_{2w} = 4 \cdot k_b \cdot \sqrt{n_c} \tag{a.12}$$

Und für das Durchsuchen einer Fünfergruppe

$$K_{2s} = 2 \cdot k_s \cdot \sqrt{n_c} \tag{a.13}$$

Die Kosten der gewöhnlichen Version von CSD und der erweiterten Version von CSD sind also gleich, wenn

$$
\begin{aligned}
K_{l,ges} &= K_{ls,ges} + K_{lw,ges} \\
&= b \cdot k_b \cdot \sqrt{n_c} + s \frac{n}{5n_c} \cdot 5 \cdot k_s \cdot \sqrt{n_c} \\
&= b \cdot 4 \cdot k_b \cdot \sqrt{n_c} + s \frac{n}{5n_c} \cdot 2 \cdot k_s \cdot \sqrt{n_c} \\
&= K_{2s,ges} + K_{2w,ges} \\
&= K_{2,ges}
\end{aligned}
\qquad \text{(a.14)}
$$

Wie in 4.3.2 gibt n/n_c die Anzahl der Zellen und damit $n/(5n_c)$ die Anzahl der Fünfergruppen an. Stellt man Gl. (a.14) um, so sieht man, dass für

$$
\frac{n}{5n_c} \cdot \frac{s}{b} \cdot \frac{k_s}{k_b} > 1 \qquad \text{(a.15)}
$$

die erweiterte Version von CSD effizienter ist die gewöhnliche Version von CSD. Einfluss haben also die Anzahl der Fünfergruppen, das Verhältnis der Anzahl der Suchen zu Bekanntmachungen und das Verhältnis der Kosten von Suchen zu Bekanntmachungen. Die erweiterte Version von CSD senkt die Kosten von Dienstsuchen und nimmt dafür einen erhöhten Aufwand für Bekanntmachungen in Kauf. Wie erwartet, ist ein Einsatz der erweiterten Version von CSD demnach um so sinnvoller, je größer die Anzahl der Fünfergruppen ist, je mehr Dienstsuchen pro Zellwechsel erfolgen und je höher die Kosten von Dienstsuchen im Vergleich Bekanntmachungen sind.

Abschließend werden einige Zahlenbeispiele betrachtet. Für Werte von $k_s/k_b=0{,}1$ wie sie auch in der Analyse in 4.3.5 verwendet wurden (der Aufwand für eine Bekanntmachung ist zehnmal so hoch wie für eine Suche), ist der Einsatz der erweiterten Version von CSD für ein Verhältnis von Suchen zu Bekanntmachungen von 1:1 ab einer Netzwerkgröße von 50 Zellen sinnvoll. Bei einem Verhältnis von Suchen zu Bekanntmachungen von 3:1 ist der Einsatz der erweiterten Version schon ab einer Netzwerkgröße von 16 Zellen sinnvoll.

A.7 Verwendete Nachrichtentypen

Hier sind die in der Simulation verwendeten Nachrichtentypen mit den wichtigsten in ihnen enthaltenen Feldern und deren Größen dargestellt.

1. Standard-Nachrichtenkopf

Der Standard-Nachrichtenkopf (*standard header*) ist in jeder versendeten Nachricht enthalten. Er legt den Nachrichtentyp, die Nachrichtenlänge sowie Ziel- und Quellknoten jeweils mit ID und Position fest.

Standard-Nachrichtenkopf	
Feldname	**Größe [Bytes]**
Message type	2
Message length	2
Destination ID	4
Destination x	4
Destination y	4
Source ID	4
Source x	4
Source y	4
Gesamt	**28**

2. Funkfeuer-Nachricht

Funkfeuer-Nachrichten werden von Nachbarknoten zum Austausch ihrer Positionen verwendet. Zusätzlich werden sie auch zur Bestimmung der Hauptknoten genutzt. Neben dem Standard-Nachrichtenkopf enthalten Funkfeuer-Nachrichten daher die Identität und Position des aktuellen Hauptknotens (soweit bekannt), ein Hinweis, ob der mit dieser Nachricht übermittelte Hauptknoten zwangsweise übernommen werden muss (zur Auflösung von Konkurrenzsituationen zwischen Hauptknoten), und einen Zeitstempel, um den Ausfall von Hauptknoten detektieren zu können.

Funkfeuer	
Feldname	**Größe [Bytes]**
Standard header	28
Master ID	4
Master x	4
Master y	4
Force switch	1
Time stamp	8
Gesamt	**49**

3. Bekanntmachung

Bekanntmachungen werden von gewöhnlichen Knoten versendet, um die von ihnen angebotenen Dienste bei ihren jeweiligen Hauptknoten zu registrieren. Sie enthalten neben dem Standard-Nachrichtenkopf noch die Anzahl der übermittelten Dienstbeschreibungen und die Dienstbeschreibungen selbst.

Bekanntmachung	
Feldname	**Größe [Bytes]**
Standard header	28
No. of services	1
Service descriptions	4096
Gesamt	**4125**

4. Aktualisierung

Aktualisierungen sind eine Kurzform der Bekanntmachung, die verschickt werden, wenn ein gewöhnlicher Knoten sich innerhalb seiner Zelle bewegt hat, die von ihm angebotenen Dienste aber unverändert geblieben sind. Ebenso können Sie dem Hauptknoten bei längerer Inaktivität als Lebenszeichen dienen.

Aktualisierung	
Feldname	**Größe [Bytes]**
Standard header	28
No. of changed services	1
Gesamt	**29**

5. Übergabe-Nachricht

Übergabe-Nachrichten werden von Hauptknoten an ihre Nachfolger versendet, wenn Hauptknoten ihre Zelle verlassen wollen.

Übergabe	
Feldname	**Größe [Bytes]**
Standard header	28
Gesamt	**28**

6. Suche in Zelle

Suchnachrichten in Zellen werden von gewöhnlichen Knoten an ihre Hauptknoten gesendet, wenn sie einen Dienst suchen. Neben dem Standard-Nachrichtenkopf enthält die Suchnachricht eine ID, um Suchantworten zuordnen zu können, und eine Beschreibung des gesuchten Dienstes.

Suche in Zelle	
Feldname	**Größe [Bytes]**
Standard header	28
Search ID	1
Service description	512
Gesamt	**541**

7. Suche zwischen Zellen

Suchnachrichten zwischen Zellen werden von Hauptknoten ausgetauscht, wenn ein gesuchter Dienst nicht in der eigenen Zelle verfügbar ist. Neben dem Standard-Nachrichtenkopf enthält die Suchnachricht eine ID, um Suchantworten zuordnen zu können, und eine Beschreibung des gesuchten Dienstes.

Suche zwischen Zellen	
Feldname	**Größe [Bytes]**
Standard header	28
Search ID	1
Service description	512
Gesamt	**541**

8. Antwort in Zelle

Suchantworten in Zellen werden von Hauptknoten an gewöhnliche Knoten gesendet, wenn ein gewünschter Dienst lokalisiert wurde. Die Nachricht enthält eine ID, um die Suchantworten zuordnen zu können, und die Identität und Position des gefundenen Dienstleisters.

Anwort in Zelle	
Feldname	**Größe [Bytes]**
Standard header	28
Search ID	1
Service Provider ID	4
Service Provider x	4
Service Provider y	4
Gesamt	**41**

9. Antwort zwischen Zellen

Suchantworten zwischen Zellen werden von Hauptknoten an andere Zellen gesendet, wenn ein gesuchter Dienst in der eigenen Zelle verfügbar ist. Die Nachricht enthält eine ID, um die Suchantworten zuordnen zu können, und Identität und Position des Dienstleisters.

Anwort zwischen Zellen	
Feldname	**Größe [Bytes]**
Standard header	28
Search ID	1
Service Provider ID	4
Service Provider x	4
Service Provider y	4
Gesamt	**41**

A.8 Weitere Simulationsergebnisse

Im Folgenden finden sich einige weitere Diagramme mit Simulationsergebnissen, die von Interesse sind und auf die im Text verwiesen wird, die aber aus Platzgründen nicht in Kapitel 5 enthalten sind.

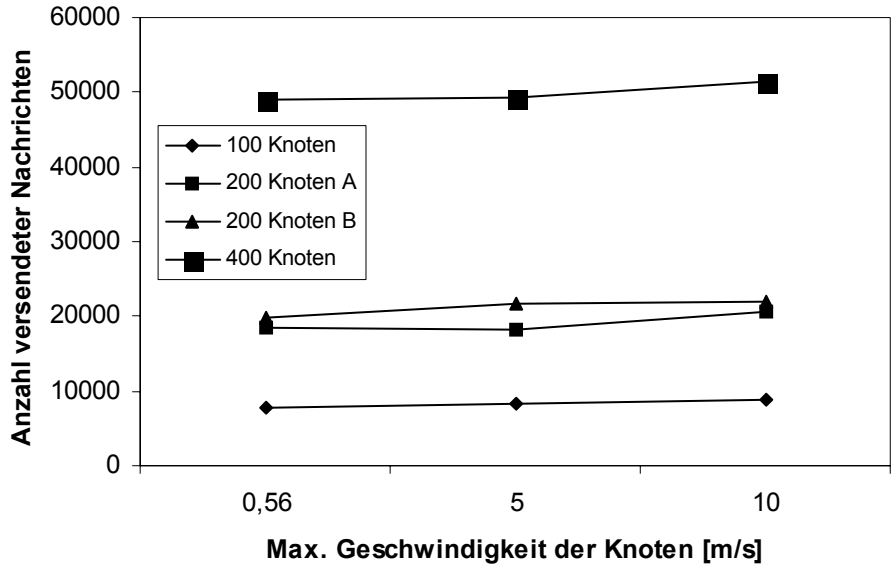

Abbildung A-2. Einfluss der Knotengeschwindigkeit auf die Gesamtzahl der Nachrichten

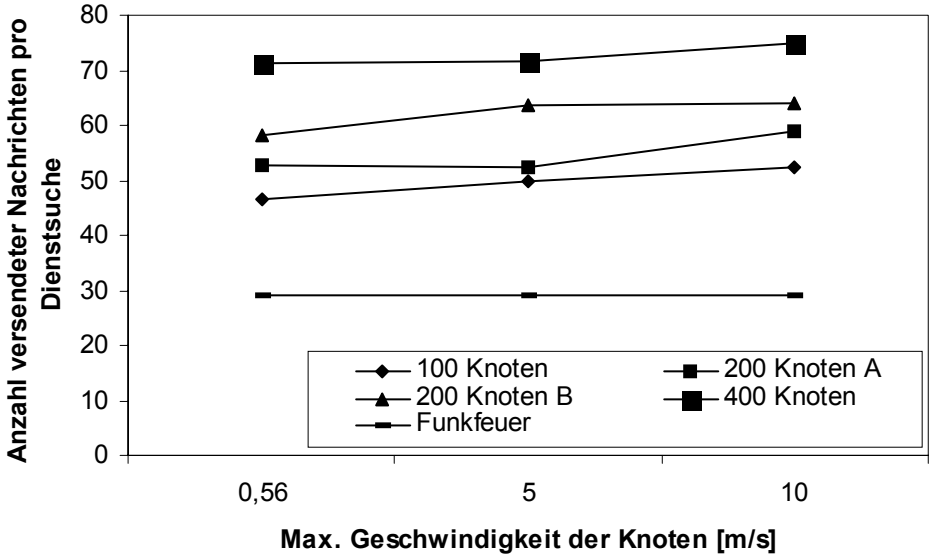

Abbildung A-3. Einfluss der Geschwindigkeit auf die Zahl der Nachrichten je Dienstsuche

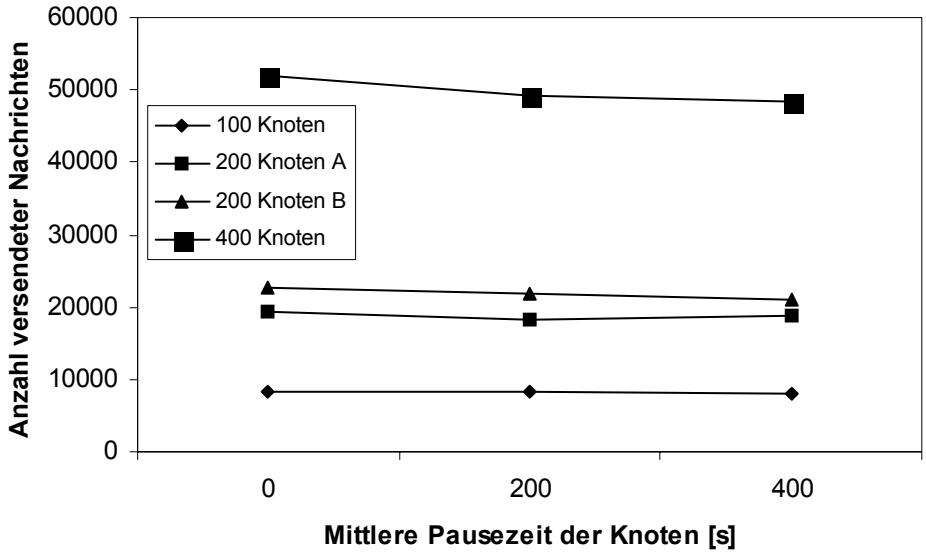

Abbildung A-4. Einfluss der Pausezeit der Knoten auf die Gesamtzahl der Nachrichten

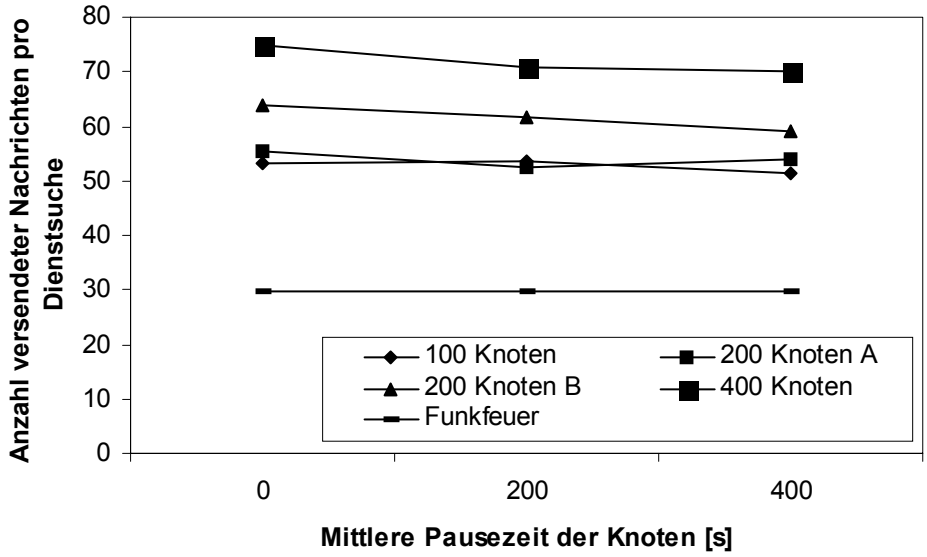

Abbildung A-5. Einfluss der Pausezeit der Knoten auf die Zahl der Nachrichten je Dienstsuche

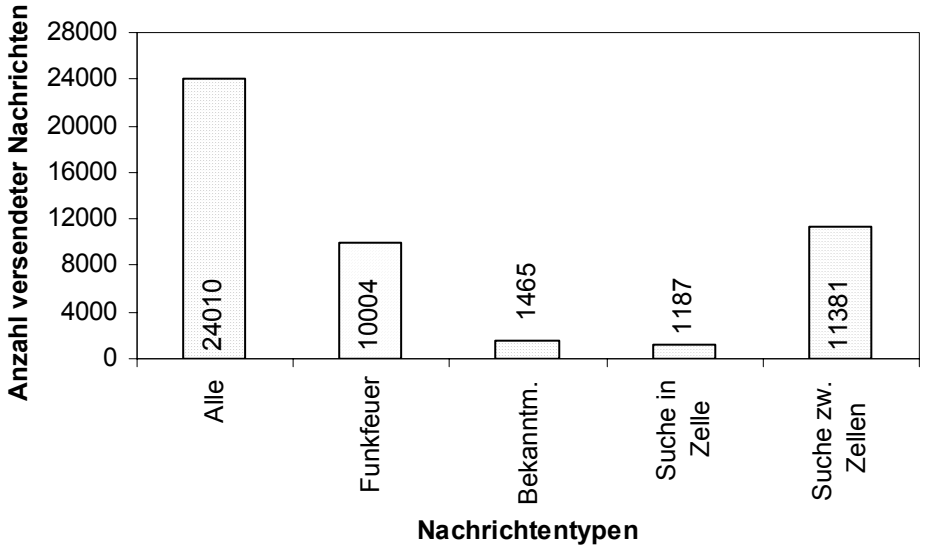

Abbildung A-6. Aufschlüsselung der Anzahl der versendeten Nachrichten nach Typ für 200 Knoten in 9 Zellen und einer erhöhten Anfragerate von 1 Anfrage/30s

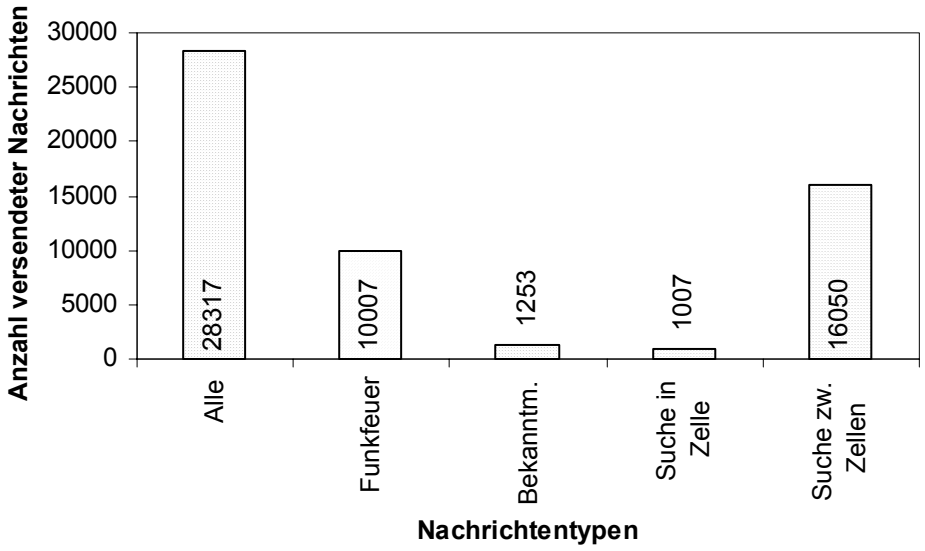

Abbildung A-7. Aufschlüsselung der Anzahl der versendeten Nachrichten nach Typ für 200 Knoten in 16 Zellen und einer erhöhten Anfragerate von 1 Anfrage/30s

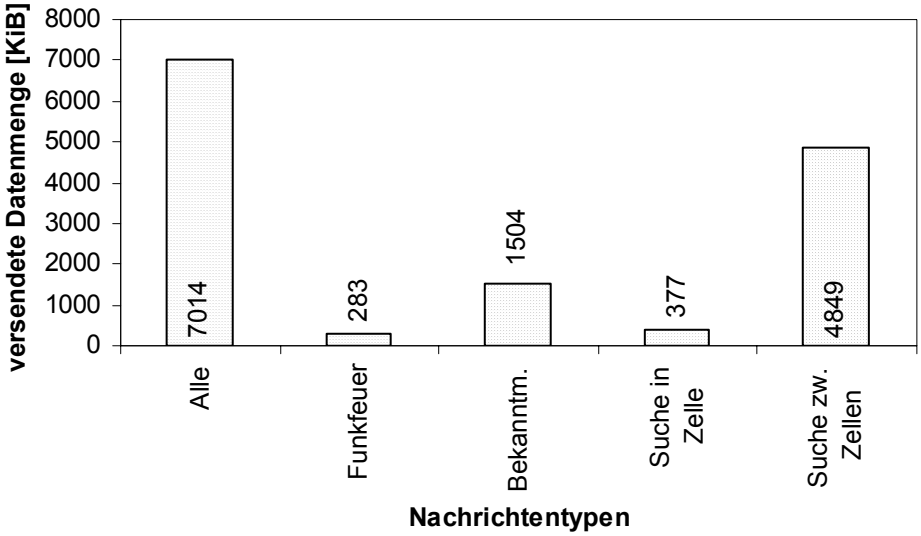

Abbildung A-8. Aufschlüsselung der versendeten Datenmenge nach Nachrichtentyp für 200 Knoten in 9 Zellen und einer erhöhten Anfragerate von 1 Anfrage/30s

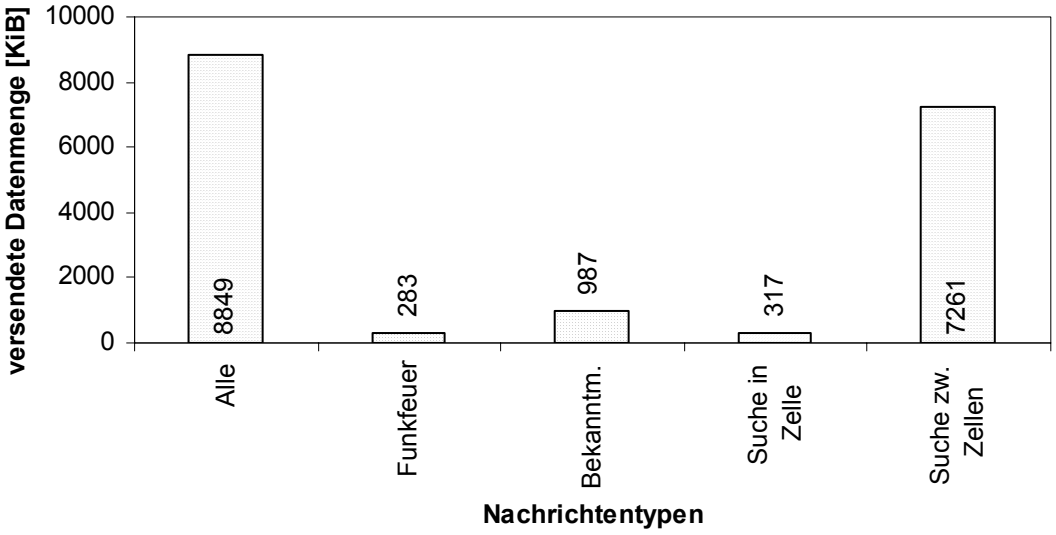

Abbildung A-9. Aufschlüsselung der versendeten Datenmenge nach Nachrichtentyp für 200 Knoten in 16 Zellen und einer erhöhten Anfragerate von 1 Anfrage/30s

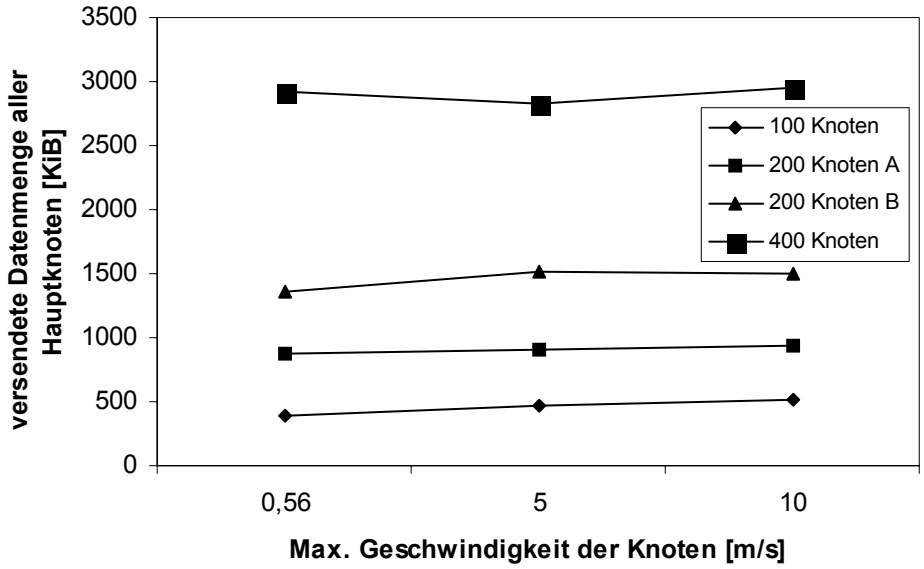

Abbildung A-10. Einfluss der Geschwindigkeit auf versendete Datenmenge aller Hauptknoten

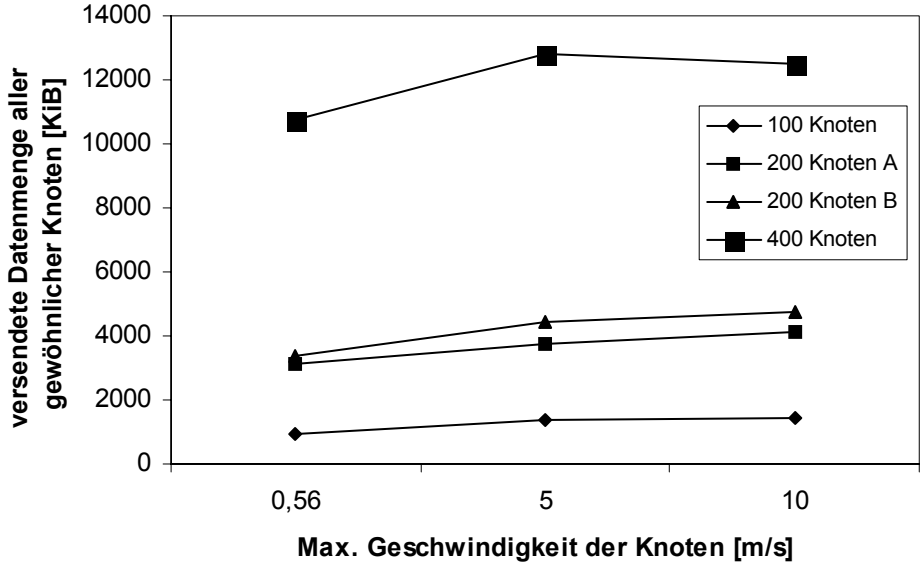

Abbildung A-11. Einfluss der Geschwindigkeit auf die versendete Datenmenge aller gewöhnlichen Knoten

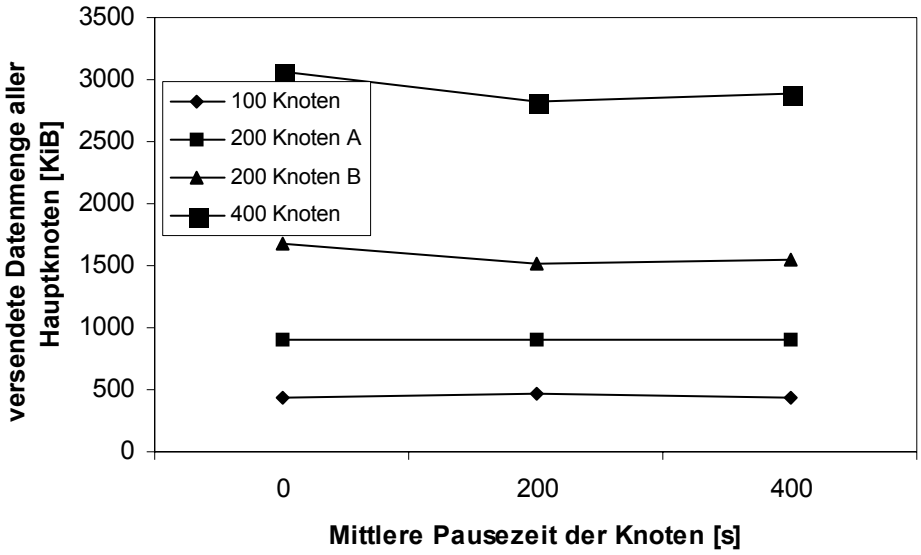

Abbildung A-12. Einfluss der mittleren Pausezeit auf die versendete Datenmenge aller Hauptknoten

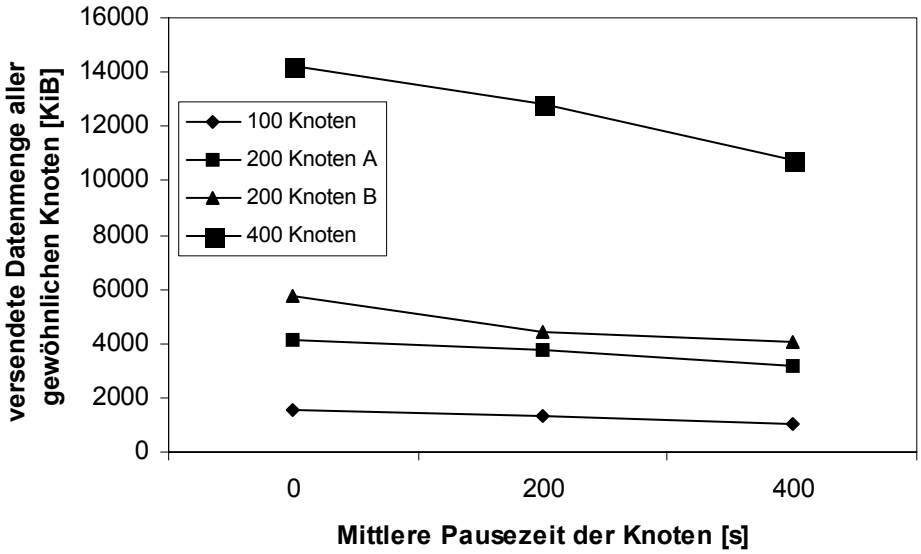

Abbildung A-13. Einfluss der mittleren Pausezeit auf die versendete Datenmenge aller gewöhnlichen Knoten

A.9 Integration und Nutzung der Dienst-Entdeckung auf dem Khepera

Der Programmcode für die Dienst-Entdeckung verteilt sich auf insgesamt vier Dateien. sd.c, global_sd.c sowie die zugehörigen Header-Dateien sd.h und global_sd.h. In sd.c sind sämtliche Funktionen gekapselt, die für die Dienst-Entdeckung benötigt werden und diese direkt betreffen. In global_sd.c finden sich wiederum einige allgemeine Funktionen, die benötigt werden, beispielsweise für die Bearbeitung von Strings, aber nicht die Dienst-Entdeckung an sich betreffen. Weiterhin werden hier auch die Dienstbeschreibungen verwaltet. global_sd.c kann vom Nutzer modifiziert werden, ohne Details der Implementierung der Dienst-Entdeckung zu kennen. Beispielsweise um Dienstbeschreibungen hinzuzufügen oder Funktionen einzufügen, die allen Knoten, die die Dienst-Entdeckung verwenden, zur Verfügung stehen sollen. Die Integration der Dienst-Entdeckung in ein Benutzerprogramm erfolgt in vier Schritten.

1. In global_sd.h werden die im Netzwerk verfügbaren Dienste beschrieben. Da zurzeit keine semantische Suche angewandt wird, können hier beispielsweise einfach eindeutige IDs genutzt werden.

```
//define services
#define CAMERA_SERVICE 10
#define GRIPPER_SERVICE 15
```

2. Jeder Roboter, der an der Dienst-Entdeckung teilnehmen möchte, bindet beide Header-Dateien ein.

```
#include "global_sd.h"
#include "sd.h"
```

3. Vor einer Nutzung ist eine Initialisierung der Dienst-Entdeckung erforderlich. Ebenso müssen die Threads gestartet werden, die für die Dienst-Entdeckung nötig sind.

```
//initialize tables, timers, …
initSD();

//start threads
 //start receiverThread
 status = install_task ("receiver", 4800, receiverThread);
 if (status == -1)
   exit (0);
 vIDProcess[1] = (uint32) status;

 //start updateThread
 status = install_task ("update", 4800, updateThread);
 if (status == -1)
   exit (0);
 vIDProcess[2] = (uint32) status;
```

4. Es ist darauf zu achten, dass das zugrunde liegende Bluetooth-Kommunikationssystem die Dienst-Entdeckung benachrichtigt, wenn sich der Master eines Knoten geändert hat. Hierfür ist die Funktion *masterChanged* vorgesehen.

```
//master is now robot with ID newMasterID
masterChanged(newMasterID);
```

Nun kann die Dienst-Entdeckung genutzt werden. Das Hinzufügen und Entfernen von Diensten erfolgt über die Funktionen *addService* und *deleteService*.

```
//add and delete own services
addService(CAMERA_SERVICE);
deleteService(CAMERA_SERVICE);
```

Die Suche nach einem Dienst erfolgt über die Funktion *searchService*.

```
//search a service
searchService(CAMERA_SERVICE);
while (!serviceSearchCompleted())
  tim_suspend_task(1000); //sleep 1 second
printf("found service provider %d\n\r", getServiceProviderID());
```

Glossar / Abkürzungsverzeichnis

Anycast. Anycast bezeichnet das Versenden einer Nachricht an einen beliebigen Computer aus einer Gruppe von Computern. Für den Sender erfolgt die Auswahl transparent. Bei Verwendung des IP-Protokolls sendet er beispielsweise einfach ein IP-Paket an eine Anycast IP-Adresse, die sich strukturell nicht von anderen IP-Adressen unterscheidet. Welcher Computer aus der Rechnergruppe nun tatsächlich diese Nachricht erhält, wird durch den verwendeten Routing-Algorithmus bestimmt. Bei Verwendung von Anycast kann es passieren, dass aufeinander folgende Pakete an unterschiedliche Computer der Gruppe geleitet werden. Daher wird Anycast vorzugsweise bei verbindungslosen Protokollen eingesetzt.

Broadcast. Broadcast bezeichnet das Versenden einer Nachricht an alle Teilnehmer eines Netzwerks oder alle Teilnehmer innerhalb der Sendereichweite eines Funksenders.

CSD. Das in dieser Arbeit entwickelte Cell-based Service Discovery-Protokoll.

Dienst. Ein Dienst führt Funktionen für den Nutzer dieses Dienstes aus. Dieser Vorgang wird auch als das Erbringen einer Dienstleistung bezeichnet. Ein Dienst kann mehrere verschiedene Dienstleistungen anbieten. Eine Einheit, die einen oder auch mehrere solcher Dienste zur Verfügung stellt, wird als Dienstleister oder auch Dienstanbieter bezeichnet. Die Einheit für den der Dienst die Funktion ausführt, ist der Dienstnutzer oder Kunde. Die Lokalisierung eines Dienstes und des zugehörigen Dienstleisters ist die Dienst-Entdeckung.

Dienstanbieter. Siehe Dienst.

Dienstleister. Siehe Dienst.

Dienstleistung. Siehe Dienst.

Dienstnutzer. Siehe Dienst.

Einzel-Hop-Verbindung. Einzel-Hop-Verbindungen sind Kommunikationsverbindungen, bei denen ein zu übermittelndes Nachrichtenpaket durch nur einmaliges Senden direkt von der Quelle zum Ziel transferiert werden kann.

Fluten. Fluten bezeichnet das Versenden einer Nachricht an alle Teilnehmer eines Netzwerks.

GCLP. Geography-based Content Location Protocol. Ein Protokoll für das Entdecken von Diensten in mobilen Ad-hoc-Netzwerken. Siehe 2.2.2.4.

GLS. Grid Location Service. Ein Protokoll für das Entdecken von Kommunikationsknoten in mobilen Ad-hoc-Netzwerken. Siehe 2.2.2.2.

GPSR. Greedy Perimeter Stateless Routing. Ein positionsbasiertes Routing-Protokoll. Siehe 2.1.

Knoten. Siehe Kommunikationsknoten.

Kommunikationsknoten. Hier: ein mobiles Gerät (insbesondere ein mobiler Roboter), das mit einem drahtlosen Kommunikationsmodul ausgestattet und Teil eines mobilen Ad-hoc-Netzwerks ist.

Kunde. Siehe Dienstnutzer.

LANES. Lightweight Overlay for Service Discovery in Mobile Ad-hoc Networks. Ein Protokoll für das Entdecken von Diensten in mobilen Ad-hoc-Netzwerken. Siehe 2.2.2.3.

Manhattanabstand. Der Abstand zweier Punkte berechnet aus der Summe der Beträge der Differenzen ihrer Koordinaten.

Multi-Hop-Verbindung. Multi-Hop-Verbindungen sind Kommunikationsverbindungen, bei denen ein Nachrichtenpaket mehr als einmal

versendet und von Knoten zu Knoten weitergereicht werden muss, um es von seiner Quelle zum Ziel zu transferieren.

Multicast. Multicast bezeichnet das Versenden einer Nachricht durch einen Sender an mehrere Empfänger gleichzeitig. Die Nachricht wird dafür in Verteilern (wie z. B. Switches) bei Bedarf kopiert und die Kopien separat weitergeleitet.

Repliken. Bezeichnet mehrere von der Funktionalität her gleichwertige Dienste im Netzwerk.

RR. Rendezvous Regions. Ein Protokoll für das Entdecken von Diensten in mobilen Ad-hoc-Netzwerken. Siehe 2.2.2.5.

Unicast. Unicast bezeichnet das Versenden einer Nachricht an genau ein Ziel. Hierfür werden so genannte Punkt-zu-Punkt-Verbindungen (ohne vermittelnde Zwischenstationen) oder auch Ende-zu-Ende-Verbindungen (mit vermittelnden Zwischenstationen) zwischen den beiden beteiligten Netzwerkteilnehmern aufgebaut.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[AK05] H. Alex, M. Kumar and B. A. Shirazi, "Service Discovery in Wireless and Mobile Networks," in Wireless Information Highways, chapter 8, Idea Group Publication, 2005

[AKU04] J. N. Al-Karaki, A. E. Kamal, R. Ul-Mustafa, ”On the Optimal Clustering in Mobile Ad hoc Networks,“ in Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC 2004), Las Vegas, USA, January 2004, pp. 71-76

[Amaz] Amazon. Amazon Search WSDL File [Online]. Available: soap.amazon.com/schemas2/AmazonWebServices.wsdl

[APVH00] A. D. Amis, R. Prakash, T.H.P. Vuong and D.T. Huynh, “Max-Min D-Cluster Formation in Wireless Ad Hoc Networks,” in Proceedings of the 19th IEEE Conference on Computer Communications (INFOCOM 2000), Tel Aviv, Israel, March 2000, pp.32-41

[BB03] R. Beraldi, R. Baldoni, "A Caching Scheme for Routing in Mobile Ad hoc Networks and its Application to ZRP," in IEEE Transactions on Computers, Vol. 52, No. 8, August 2003, pp. 1051 - 1062

[BDT99] E. Bonabeau, M. Dorigo, G. Theraulaz, “Swarm Intelligence: From Natural to Artificial Systems,” Oxford University Press, 1999

[BKKM+03] H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, I. Stoica, "Looking Up Data in P2P Systems", in Communications of the ACM, Vol. 46, No. 2, February 2003, pp. 43-48

[BKS94] A. Bar-Noy, I. Kessler, M. Sidi, "Mobile users: to update or not to update?," in Proceedings of the 13th IEEE Conference on Computer Communications (INFOCOM ‘94), Toronto, Ontario, Canada, June 12-16, 1994, pp. 570-576

[BM04] J. Bohn, F. Mattern, “Super-Distributed RFID Tag Infrastructures,” in Proc. of the 2nd European Symposium on Ambient Intelligence (EUSAI 2004), Eindhoven, The Netherlands, November 2004, pp. 1-12

[Bosc06] K. Bosch, "Elementare Einführung in die Wahrscheinlichkeitsrechnung," Vieweg Verlag, 9th edition, 2006

[BSIG03] Bluetooth Special Interest Group (2003). Bluetooth Core Specification v1.2 [Online]. Available: https://www.bluetooth.org/spec/

[CAM+03] J. A. Crisp, M. Adler, J. R. Matijevic, S. W. Squyres, R. E. Arvidson, D. M. Kass, "Mars Exploration Rover Mission," in Journal of Geophysical Research, Vol. 108, No. E12, 8061, 2003

[Camp05] T. Camp, "Location Information Services in Mobile Ad Hoc Networks," in Handbook of Algorithms for Mobile and Wireless Networking and Computing, chapter 14, Chapman & Hall/CRC, 2005, pp. 317-339

[CH05] Z. Cheng and W. Heinzelman, "Flooding Strategy for Target Discovery in Wireless Networks," in ACM/Baltzer Wireless Networks, Vol. 11, No. 5, September 2005, pp. 607-618

[CR03] Y. Chawathe, S. Ratnasamy, L. Breslau, S. Shenker, "Making Gnutella-like P2P Systems Scalable," in Proc. of the ACM/SIGCOMM Conference on Communication Architectures, Protocols and Applications, Karlsruhe, Germany, August 2003, pp. 407-418

[CS06] H. Chesbrough and J. Spohrer, “A Research Manifesto for Services Science,” in Communications of the ACM, Vol. 49, No. 7, July 2006, pp. 35-40

[CW04] G. Cao, J. L. Welch, "Accurate Multihop Clock Synchronization in Mobile Ad Hoc Networks," in Proceedings of the 2004 International Conference on Parallel Processing Workshops (ICPPW'04), Montreal, Canada, August 2004, pp. 13-20

[DB03] J. L. Du, T. Bräunl, "Collaborative Cube Clustering with Local Image Processing," in Proceedings of the 2nd International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2003), February 2003, Brisbane, Australia, pp. 247-248

[DB04] G. Ding, B. Bhargava, "Peer-to-peer File-sharing over Mobile Ad hoc Networks," in Proc. of the 2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW 2004), Orlando, Florida, USA, March 2004

[DWR05a] J. L. Du, S. Rührup, U. Witkowski, U. Rückert, "Resource and Service Discovery for Large-Scale Robot Networks in Disaster Scenarios," in Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR2005), June 2005, Kobe, Japan, pp. 7-12

[DWR05b] J. L. Du, U. Witkowski, U. Rückert, "CSD: Cell-based Service Discovery in Large-Scale Robot Networks" in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), August 2005, Edmonton, Alberta, Canada, pp. 2235 - 2240

[DWR05c] J. L. Du, U. Witkowski, U. Rückert, “Teleoperation of a Mobile Autonomous Robot using Web Services” in Proceedings of the 3rd International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2005), September 2005, Fukui, Japan, pp. 55-60

[EGE02] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in ACM SIGOPS Operating Systems Review, Vol. 36, Issue SI, 2002, pp. 147-163

[Enge04] R. van Engelen, "Code Generation Techniques for Developing Light-weight XML Web Services for Embedded Devices" in Proc. of the 2004 ACM Symposium on Applied Computing, Nicosia, Cyprus, March 2004, pp. 854-861

[GGK01] P. Gupta, R. Gray, P. R. Kumar, “An Experimental Scaling Law for Ad Hoc Networks,” Technical Report, University of Illinois at Urbana-Champaign, USA, May 2001

[GK00] P. Gupta, P. R. Kumar, “The Capacity of Wireless Networks,” in IEEE Transactions on Information Theory, Vol. 46, No. 2, March 2000, pp. 388-404

[GM04] B. P. Gerkey, M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems," in International Journal of Robotics Research, Vol. 23, No. 9, September 2004, pp.939-954

[GLR05] M. Gerla, C. Lindemann, A. Rowstron, "P2P MANETs - New Research Issues," in Perspectives Workshop: Peer-to-Peer Mobile Ad Hoc Networks - New Research Issues, Proceedings of the Dagstuhl Seminar, Dagstuhl, Germany, July 2005

[Goog] Google. Google Search WSDL File [Online]. Available: api.google.com/GoogleSearch.wsdl

[Gros04] M. Grosseschallau, "Implementation of an ASCII Interface to the BlueCore2-External Bluetooth Module for the Minirobot Khepera", Studienarbeit, System and Circuit Technology Research Group, University of Paderborn, Germany, September 2004

[GS69] K. Gabriel, R. Sokal, "A New Statistical Approach to Geographic Variation Analysis," in Systematic Zoology, Vol. 18, 1969, pp. 259-278

[GSB04] S. Giordano, I. Stojmenovic, L. Blazevic, "Position based routing algorithms for ad hoc networks: A taxonomy," in Ad Hoc Wireless Networking, Kluwer Academic Publishers, 2004, pp. 103-136

[HC02] R. M. Hambly, T. A. Clark, "Critical Evaluation of the Motorola M12+ GPS Timing Receiver vs. the Master Clock at the United States Naval Observatory," in Proceedings of the 34th Annual Precise Time and Time Interval Systems and Applications Meeting (PTTI 2002), December 2002, Reston, Virginia, USA

[Helm05] A. Helmy, "Efficient Resource Discovery in Wireless Ad Hoc Networks: Contacts Do Help," in Resource Management in Wireless Networking, Vol. 16, book chapter, Springer, 2005

[HP02] Z. J. Haas, M. R. Pearlman, P. Samar (July 2002). The Zone Routing Protocol (ZRP) for Ad Hoc Networks, IETF Internet Draft [Online]. Available: http://www3.ietf.org/proceedings/02nov/I-D/draft-ietf-manet-zone-zrp-04.txt

[HPS06] A. Howard, L. E. Parker, G. S. Sukhatme, “Experiments with a Large Heterogeneous Mobile Robot Team: Exploration, Mapping, Deployment and Detection,” in International Journal of Robotics Research, Vol. 25, No. 5-6, 2006, pp.431-447

[HSC05] Y.-G. Ha, J.-C. Sohn and Y.-J. Cho, "Service-Oriented Integration of Networked Robots with Ubiquitous Sensors and Devices Using the Semantic Web Services Technology," in Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), Edmonton, Alberta, Canada, August 2005

[JBAS05] A. Jardosh, E. M. Belding-Royer, K. C. Almeroth, S. Suri, "Real world Environment Models for Mobile Ad hoc Networks," in IEEE Journal on Special Areas in Communications, Special Issue on Wireless Ad hoc Networks, January 2005

[JD96] D. Johnson, D. Maltz, "Dynamic source routing in ad hoc wireless networks," in Mobile Computing, Vol. 353, Kluwer Academic Publishers, 1996

[JZ00] D. Jung, A. Zelinsky, "Grounded symbolic communication between heterogeneous cooperating robots," in Autonomous Robots, Vol. 8, No. 3, July 2000, pp. 269-292

[KFOA+04] K. Konolige, D. Fox, C. Ortiz, A. Agno, M. Eriksen, B. Limketkai et al., "Centibots: Very large scale distributed robotic teams," in Proc. of the 9th International Symposium on Experimental Robotics 2004 (ISER 2004), Singapore, June 2004

[KGKS05] Y.-J. Kim, R. Govindan, B. Karp, S. Shenker, “Geographic Routing Made Practical,” in Proc. of the 2nd Symposium on Networked Systems Design and Implementation (NSDI 2005), Boston, MA, USA, May 2005

[KK00] B. Karp, H. T. Kung, “Greedy Perimeter Stateless Routing for Wireless Networks,” in Proc. of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000), Boston, MA, USA, August 2000, pp. 243-254

[KKO03] M. Klein, B. König-Ries, P. Obreiter, “Lanes - A Lightweight Overlay for Service Discovery in Mobile Ad Hoc Network,” in Proc. of the 3rd Workshop on Applications and Services in Wireless Networks (ASWN2003), Berne, Switzerland, July 2003

[KMP99] G. Karumanchi, S. Muralidharan, R. Prakash, "Information Dissemination in Partitionable Mobile Ad Hoc Networks," in Proceedings of the 19th IEEE Symposium on Reliable Distributed Systems, Lausanne, Switzerland, October 20-22, 1999, pp.4-13

[KTEA] K-Team. Khepera II Robot [Online]. Available: http://www.k-team.com

[KTNM+99] H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, T. Takahashi, A. Shinjou et al., "RoboCup Rescue: search and rescue in large-scale disasters as a domain for autonomous agents research," in Proc. of the IEEE International Conference on Systems, Man, and Cybernetics (SMC '99), Vol. 6, October 1999, pp. 739-743

[LBCL+01] J. Li, C. Blake, D. S. J. De Couto, H. I. Lee, R. Morris, “Capacity of Ad Hoc Wireless Networks,” in Proc. of the 7th Annual International Conference on Mobile Computing and Networking (MobiCom '01), Rome, Italy, July 2001, pp. 61-69

[LGMV05] M. Long, A. Gage, R. Murphy, K. Valavanis, "Application of the Distributed Field Robot Architecture to a Simulated Demining Task," in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2005), Barcelona, Spain, April 2005, pp. 3193-3200

[LJCK+00] J. Li, J. Jannotti, D. S. J. De Couto, D. R. Karger, R. Morris, “A scalable location service for geographic ad hoc routing,” in Proc. of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000), Boston, MA, USA, August 2000, pp. 120-130

[LMP03] M. T. Long, R. R. Murphy, L. E. Parker, “Distributed Multi-Agent Diagnosis and Recovery from Sensor Failures,” in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), Las Vegas, Nevada, USA, October 2003, pp. 2506-2513

[LR02] A. Lankenau, T. Röfer, “Mobile Robot Self-Localization in Large-Scale Environments,” in Proc. of the 2002 IEEE International Conference on Robotics and Automation (ICRA 2002), Washington D.C., USA, May 2002, pp. 1359-1364

[LYNG+04] J. Liu, Y. Yuan, D. M. Nicol, R. S. Gray, C. C. Newport, D. Kotz, L. F. Perrone, “Simulation validation using direct execution of wireless ad hoc routing protocols,” in Proceedings of the 18th Workshop on Parallel and Distributed Simulation (PADS 2004), Kufstein, Austria, May 2004, pp. 7–16

[Matt93] F. Mattern, "Distributed Control Algorithms (Selected Topics)," in Parallel Computing on Distributed Memory Multiprocessors, Springer, 1993, pp. 167-185

[MF] S. McCanne and S. Floyd. ns Network Simulator [Online]. Available: http://www.isi.edu/nsnam/ns/

[MHWL03] M. Mauve, H. Füßler, J. Widmer, T. Lang, "Position-Based Multicast Routing for Mobile Ad-Hoc Networks," Technical Report TR-03-004, Department of Computer Science, University of Mannheim, Germany, 2003

[Micr06] Microsoft (October 2006). Microsoft Robotics Studio October 2006 CTP [Online]. Available: http://msdn.microsoft.com/robotics/

[MSS] S. Kurkowski, T. Camp, M. Colagrosso, “MANET Simulation Studies: The Incredibles,” in ACM Mobile Computing and Communications Review, Vol. 9, No. 4, October 2005, pp. 50-61

[MWH01] M. Mauve, J. Widmer, H. Hartenstein, "A Survey on Position-Based Routing in Mobile Ad-Hoc Networks," in IEEE Network Magazine, Vol. 15, No. 6, November 2001, pp. 30-39

[NASA] NASA Jet Propulsion Laboratory. Mars Artwork: Rover [Online]. Available: http://marsrover.nasa.gov/gallery/artwork/hires/rover1.jpg

[Newm00] J. Newmarch, “A Programmer's Guide to Jini Technology,” 1st edition, Apress, November 2000

[NG05] V. Naumov, T. Gross, "Scalability of Routing Methods in Ad Hoc Networks," in Performance Evaluation, Vol. 62, October 2005, pp. 193-209

[OASI04] OASIS Consortium (2004). UDDI Version 3 Specification, OASIS Standard [Online]. Available: http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm

[OVM05] C. L. Ortiz, R. Vincent, B. Morisset, "Task inference and distributed task management in the Centibots robotic system," in Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), Utrecht, The Netherlands, July 2005, pp.860-867

[Park98] L. E. Parker, "ALLIANCE: An Architecture for Fault Tolerant Multi-Robot Cooperation," in IEEE Transactions on Robotics and Automation, Vol. 14, No. 2, 1998, pp. 220-240

[PB94] C. Perkins and P. Bhagwat, “Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers,” in Proc. of the ACM/SIGCOMM Conference on Communication Architectures, Protocols and Applications, London, UK, October 1994, pp. 234-244

[PT06] L. E. Parker, F. Tang, "Building Multi-Robot Coalitions through Automated Task Solution Synthesis," in Proceedings of the IEEE, Vol. 94, No. 7, Special Issue on Multi-Robot Systems, July 2006, pp. 1289- 1305

[RBYY+02] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, S. Shenker, “GHT: A Geographic Hash Table for Data-Centric Storage,” in Proc. of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA), Atlanta, Georgia, September 2002, pp. 78-87

[RMCC+06] M. G. Rubinstein, I. M. Moraes, M. E. M. Campista, L. H. M. K. Costa, O. C. M. B. Duarte, “A Survey on Wireless Ad Hoc Networks,” in Proceedings of the 8th IFIP/IEEE International Conference on Mobile and Wireless Communication Networks, Santiago, Chile, August 20-25, 2006, pp. 1 – 33

[SH04a] K. Seada, A. Helmy, "Rendezvous Regions: A Scalable Architecture for Service Location and Data-Centric Storage in Large-Scale Wireless Networks,” in Proc. of the IEEE/ACM IPDPS International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN), Santa Fe, New Mexico, USA, April 2004

[SH04b] M. P. Singh, M. N. Huhns, "Service-Oriented Computing," John Wiley & Sons, 2004

[Shep96] T. J. Shepard, "A Channel Access Scheme for Large Dense Packet Radio Networks," in Proc. of the ACM Conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '96), Palo Alto, California, United States, August 1996, pp. 219-230

[SHG04] Karim Seada, Ahmed Helmy, Ramesh Govinda, “On the Effect of Localization Errors on Geographic Face Routing in Sensor Networks,” in Proc. of the IEEE/ACM 3rd International Symposium on Information Processing in Sensor Networks (IPSN 2004), Berkeley, CA, USA, April 2004, pp. 71-80

[SM05] C. Schlenoff, E. Messina, "A robot ontology for urban search and rescue," in Proceedings of the 2005 ACM Workshop on Research in Knowledge Representation for Autonomous Systems, Bremen, Germany, November 2005, pp. 27-34

[SRI] SRI International. Centibots Project [Online]. Available: http://www.ai.sri.com/centibots/pictures/index.html

[Stoj04] Ivan Stojmenovic, “Geocasting with Guaranteed Delivery in Sensor Networks,” in IEEE Wireless Communications, Vol. 11, No. 6, December 2004, pp. 29-37

[Stru06] M. Strünkmann, “Entwicklung eines Bluetooth-Scatternetzes für den Miniroboter Khepera“, Diplomarbeit, System and Circuit Technology Research Group, University of Paderborn, Germany, September 2006

[TMB01] M. Takai, J. Martin, R. Bagrodia, “Effects of wireless physical layer modeling in mobile ad hoc networks,” in Proceedings of the 2nd ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc ’01), Long Beach, USA, Oct. 2001, pp. 87–94

[TNCS02] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad hoc Network," in ACM Wireless Networks, Vol. 8, No. 2, March 2002, pp. 153-167

[Tous80] G. Toussaint, "The Relative Neighborhood Graph of a Finite Planar Set," in Pattern Recognition, Vol. 12, No. 4, 1980, pp. 261-268

[TV04] J. B. Tchakarov, N. H. Vaidya, "Efficient Content Location in Wireless Ad Hoc Networks," in Proc. of the IEEE International Conference on Mobile Data Management (MDM 2004), Berkeley, California, USA, January 2004, pp.74-87

[TDKW06] A. Tanoto, J. L. Du, T. Kaulmann, U. Witkowski, "MPEG-4-Based Interactive Visualization as an Analysis Tool for Experiments in Robotics," in Proceedings of the 2006 International Conference on Modeling, Simulation and Visualization Methods (MSV'06), June 2006, Las Vegas, Nevada, USA, pp. 186-192

[TDWR06] A. Tanoto, J. L. Du, U. Witkowski, U. Rückert, "Teleworkbench: An Analysis Tool for Multi-Robot Experiments," in Proceedings of the 19th IFIP World Computer Congress (WCC 2006), August 2006, Santiago, Chile, pp. 179-188

[TWR05] A. Tanoto, U. Witkowski, U. Rückert, “Teleworkbench: A Teleoperated Platform for Multi-Robot Experiments” in Proc. of the 3rd International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2005), Awara-Spa, Fukui, Japan, September 2005, pp.49-54

[UNEC04] United Nations Economic Commission for Europe, "2004 World Robotics Survey," Press Release ECE/STAT/04/P01, Geneva, Switzerland, October 2004

[USCG] U.S. Cost Guard Navigation Center. Global Positioning System [Online]. Available: http://www.navcen.uscg.gov/gps/default.htm

[W3C04] World Wide Web Consortium (W3C) (February 2004). Web Services Architecture, W3C Working Group Note [Online]. Available: http://www.w3.org/TR/ws-arch/

[WCDR+04] U. Witkowski, T. Chinapirom, J. L. Du, U. Rückert, O. Manolov, "Cooperating Autonomous and Mobile Minirobots in Dynamic Environments," in International Federation of Automatic Control - IFAC - DECOM-TT, October 2004, Bansko, Bulgaria, pp. 277-282

[WGDK+02] T. Weigel, J.-S. Gutmann, M. Dietl, A. Kleiner, B. Nebel, "CS Freiburg: coordinating robots for successful soccer playing," in IEEE Transactions on Robotics and Automation, Vol. 18, No. 5, October 2002, pp. 685-699

[WM06] K. Wrona, P. Maehoenen, "Stability and security in wireless cooperative networks," in Cooperation in Wireless Networks: Principles and Applications, chapter 10, Springer, 2006, pp. 313-363

[Wolf] Wolfram Research. Mathematica [Online]. Available: http://www.wolfram.com/

[XMET] XMethods. List of publicly available web services [Online]. Available: http://www.xmethods.net/

Eigene Veröffentlichungen

[DB03] J. L. Du, T. Bräunl, "Collaborative Cube Clustering with Local Image Processing," in Proceedings of the 2nd International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2003), February 2003, Brisbane, Australia, pp. 247-248

[DTMW+07] J. L. Du, A. Tanoto, E. Monier, U. Witkowski, U. Rückert, “Multi-Robotics Experiments using Mini-Robots” in Proceedings of the 3rd International Conference on Intelligent Computing and Information Systems (ICICIS 2007), March 2007, Cairo, Egypt

[DWR05a] J. L. Du, S. Rührup, U. Witkowski, U. Rückert, "Resource and Service Discovery for Large-Scale Robot Networks in Disaster Scenarios," in Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR2005), June 2005, Kobe, Japan, pp. 7-12

[DWR05b] J. L. Du, U. Witkowski, U. Rückert, "CSD: Cell-based Service Discovery in Large-Scale Robot Networks" in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), August 2005, Edmonton, Alberta, Canada, pp. 2235 - 2240

[DWR05c] J. L. Du, U. Witkowski, U. Rückert, “Teleoperation of a Mobile Autonomous Robot using Web Services” in Proceedings of the 3rd International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2005), September 2005, Fukui, Japan, pp. 55-60

[DWR07] J. L. Du, U. Witkowski, U. Rückert, “A Bluetooth Scatternet for the Khepera Robot” in Proceedings of the 4th International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2007), October 2007, Buenos Aires, Argentina, to appear

[TDKW06] A. Tanoto, J. L. Du, T. Kaulmann, U. Witkowski, "MPEG-4-Based Interactive Visualization as an Analysis Tool for Experiments in Robotics," in Proceedings of the 2006 International Conference on Modeling, Simulation and Visualization Methods (MSV'06), June 2006, Las Vegas, Nevada, USA, pp. 186-192

[TDWR06] A. Tanoto, J. L. Du, U. Witkowski, U. Rückert, "Teleworkbench: An Analysis Tool for Multi-Robot Experiments," in Proceedings of the 19th IFIP World Computer Congress (WCC 2006), August 2006, Santiago, Chile, pp. 179-188

[WCDR+04] U. Witkowski, T. Chinapirom, J. L. Du, U. Rückert, O. Manolov, "Cooperating Autonomous and Mobile Minirobots in Dynamic Environments," in International Federation of Automatic Control - IFAC - DECOM-TT, October 2004, Bansko, Bulgaria, pp. 277-282

Das Heinz Nixdorf Institut – Interdisziplinäres Forschungszentrum für Informatik und Technik

Das Heinz Nixdorf Institut ist ein Forschungszentrum der Universität Paderborn. Es entstand 1987 aus der Initiative und mit Förderung von Heinz Nixdorf. Damit wollte er Ingenieurwissenschaften und Informatik zusammenzuführen, um wesentliche Impulse für neue Produkte und Dienstleistungen zu erzeugen. Dies schließt auch die Wechselwirkungen mit dem gesellschaftlichen Umfeld ein.

Die Forschungsarbeit orientiert sich an dem Programm „Dynamik, Mobilität, Vernetzung: Auf dem Weg zu den technischen Systemen von morgen“. In der Lehre engagiert sich das Heinz Nixdorf Institut in vielen Studiengängen der Universität. Hier ist das übergeordnete Ziel, den Studierenden die Kompetenzen zu vermitteln, auf die es in der Wirtschaft morgen ankommt.

Heute wirken am Heinz Nixdorf Institut sieben Professoren mit insgesamt 200 Mitarbeiterinnen und Mitarbeitern. Etwa ein Viertel der Forschungsprojekte der Universität Paderborn entfallen auf das Heinz Nixdorf Institut und pro Jahr promovieren hier etwa 30 Nachwuchswissenschaftlerinnen und Nachwuchswissenschaftler.

Heinz Nixdorf Institute – Interdisciplinary Research Centre for Computer Science and Technology

The Heinz Nixdorf Institute is a research centre within the University of Paderborn. It was founded in 1987 initiated and supported by Heinz Nixdorf. By doing so he wanted to create a symbiosis of computer science and engineering in order to provide critical impetus for new products and services. This includes interactions with the social environment.

Our research is aligned with the program “Dynamics, Mobility, Integration: Enroute to the technical systems of tomorrow.” In training and education the Heinz Nixdorf Institute is involved in many programs of study at the University of Paderborn. The superior goal in education and training is to communicate competencies that are critical in tomorrows economy.

Today seven Professors and 200 researchers work at the Heinz Nixdorf Institute. The Heinz Nixdorf Institute accounts for approximately a quarter of the research projects of the University of Paderborn and per year approximately 30 young researchers receive a doctorate.

Stand: April 2006

Bände der HNI-Verlagsschriftenreihe

Bd. 1 FAHRWINKEL, U.: Methoden zur Modellierung und Analyse von Geschäftsprozessen zur Unterstützung des Business Process Reengineering. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 1, 1995 – ISBN 3-931466-00-0

Bd. 2 HORNBOSTEL, D.: Methode zur Modellierung der Informationsverarbeitung in Industrieunternehmen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 2, 1995 – ISBN 3-931466-01-9

Bd. 3 STEMANN, V.: Contention Resolution in Hashing Based Shared Memory Simulations. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 3, 1995 – ISBN 3-931466-02-7

Bd. 4 KETTERER, N.: Beschreibung von Datenaustausch eines verteilten Fertigungssteuerungssystems. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 4, 1995 – ISBN 3-931466-03-5

Bd. 5 HARTMANN, T.: Spezifikation und Klassifikation von Methoden zur Definition hierarchischer Abläufe. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 5, 1995 – ISBN 3-931466-04-3

Bd. 6 WACHSMANN, A.: Eine Bibliothek von Basisdiensten für Parallelrechner: Routing, Synchronisation, gemeinsamer Speicher. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 6, 1995 – ISBN 3-931466-05-1

Bd. 7 GAUSEMEIER, J. (Hrsg.): Die Szenario-Technik – Werkzeug für den Umgang mit einer multiplen Zukunft. 1. Paderborner Szenario-Workshop, 14. November 1995, HNI-Verlagsschriftenreihe, Paderborn, Band 7, 1995 – ISBN 3-931466-06-X

Bd. 8 CZUMAJ, A.: Parallel Algorithmic Techniques: PRAM Algorithms and PRAM Simulations. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 8, 1995 – ISBN 3- 931466-07-8

Bd. 9 HUMPERT, A.: Methodische Anforderungsverarbeitung auf Basis eines objektorientierten Anforderungsmodells. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 9, 1995 – ISBN 3-931466-08-6

Bd. 10 AMEUR, F.: Space-Bounded Learning Algorithms. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 10, 1995 –ISBN 3-931466-09-4

Bd. 11 PAUL, M.: Szenariobasiertes Konzipieren neuer Produkte des Maschinenbaus auf Grundlage möglicher zukünftiger Technologieentwicklungen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 11, 1996 – ISBN 3-931466-10-8

Bd. 12 HOLL, F.: Ordnungsmäßigkeit von Informations- und Kommunikationssystemen. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 12, 1996 – ISBN 3-931466-11-6

Bd. 13 GAUSEMEIER, J. (Hrsg.): First European Workshop on Global Engineering Networking - organized by GLENnet e.V., HNI-Verlagsschriftenreihe, Paderborn, Band 13, 1996 – ISBN 3-931466-12-4

Bd. 14 PETRI, K.: Vergleichende Untersuchung von Berechnungsmodellen zur Simulation der Dynamik von Fahrleitung-Stromabnehmer-Systemen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 14, 1996 – ISBN 3-931466-13-2

Bd. 15 LESCHKA, S.: Fallbasiertes Störungsmanagement in flexiblen Fertigungssystemen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 15, 1996 – ISBN 3-931466-14-0

Bd. 16 SCHNEIDER, U.: Ein formales Modell und eine Klassifikation für die Fertigungssteuerung - Ein Beitrag zur Systematisierung der Fertigungssteuerung. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 16, 1996 – ISBN 3-931466-15-9

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 17 FELSER, W.: Eine Methode zur Erstellung von Fertigungssteuerungsverfahren aus Bausteinen. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 17, 1996 – ISBN 3-931466-16-7

Bd. 18 GAUSEMEIER, J.; ALEXANDER FINK, A.: Neue Wege zur Produktentwicklung – Erfolgspotentiale der Zukunft. HNI-Verlagsschriftenreihe, Paderborn, Band 18, 1996– ISBN 3-931466-17-5

Bd. 19 DANGELMAIER, W.; GAUSEMEIER, J.: Fortgeschrittene Informationstechnologie in der Produktentwicklung und Fertigung. 2. Internationales Heinz Nixdorf Symposium, HNI-Verlagsschriftenreihe, Paderborn, Band 19, 1996 – ISBN 3-931466-18-3

Bd. 20 HÜLLERMEIER, E.: Reasoning about Systems based on Incomplete and Uncertain Models. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 20, 1997 – ISBN 3-931466- 19-1

Bd. 21 GAUSEMEIER, J.: International Symposium on Global Engineering Network - Antwerb, Belgium, HNI-Verlagsschriftenreihe, Paderborn, Band 21, 1997 – ISBN 3- 931466-20-5

Bd. 22 BURGER, A.: Methode zum Nachweis der Wirtschaftlichkeit von Investitionen in die Rechnerintegrierte Produktion. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 22, 1997 – ISBN 3-931466-21-3

Bd. 23 GAUSEMEIER, J.: Entwicklung und Transfer von Entwicklungssystemen der Mechatronik - Paderborner Workshop TransMechatronik. HNI-Verlagsschriftenreihe, Paderborn, Band 23, 1997 – ISBN 3- 931466-22-1

Bd. 24 GERDES, K.-H.: Architekturkonzeption für Fertigungsleitsysteme der flexiblen automatischen Fertigung. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 24, 1997 – ISBN 3-931466-23-X

Bd. 25 EBBESMEYER, P.: Dynamische Texturwände - Ein Verfahren zur echtzeitorientierten Bildgenerierung für Virtuelle Umgebungen technischer Objekte. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 25, 1997 – ISBN 3-931466-24-8

Bd. 26 FRANK, G.: Ein digitales Hardwaresystem zur echtzeitfähigen Simulation biologienaher neuronaler Netze. Dissertation, Fachbereich für Elektrotechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 26, 1997 – ISBN 3-931466-25-6

Bd. 27 DITTRICH, W.: Communication and I/O Efficient Parallel Data Structures. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 27, 1997 – ISBN 3-931466-26-4

Bd. 28 BÄUMKER, A.: Communication Efficient Parallel Searching. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 28 1997 – ISBN 3- 931466-27-2

Bd. 29 PINTASKE, C.: System- und Schaltungstechnik neuronaler Assoziativspeicher. Dissertation, Fachbereich für Elektrotechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 29, 1997 – ISBN 3-931466-28-0

Bd. 30 HENKEL, S.: Ein System von Software-Entwurfsmustern für die Propagation von Ereignissen in Werkzeugen zur kooperativen Fabrikmodellierung. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 30, 1997 – ISBN 3-931466-29-9

Bd. 31 DANGELMAIER, W.: Vision Logistik – Logistik wandelbarer Produktionsnetze. HNI-Verlagsschriftenreihe, Paderborn, Band 31, 1997 – ISBN 3-931466-30-2

Bd. 32 BREXEL, D.: Methodische Strukturmodellierung komplexer und variantenreicher Produkte des integrativen Maschinenbaus. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 32, 1997 – ISBN 3-931466-31-0

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 33 HAHN, A.: Integrationsumgebung für verteilte objektorientierte Ingenieursysteme. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 33, 1997 – ISBN 3-931466-32-9

Bd. 34 SABIN, A.: Semantisches Modell zum Aufbau von Hilfsorientierungsdiensten in einem globalen Engineering Netzwerk. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 34, 1997 – ISBN 3-931466-33-7

Bd. 35 STROTHMANN, W.-B.: Bounded Degree Spanning Trees. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 35, 1997 – ISBN 3-931466-34-5

Bd. 36 MÜLLER, W.; RAMMIG, F.-J.: Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen. HNI-Verlagsschriftenreihe, Paderborn, Band 36, 1998 – ISBN 3-931466- 35-3

Bd. 37 SCHNEIDER, W.: Anwenderorientierte Integration von CAE-Systemen. Ein Verfahren zur Realisierung eines durchgehenden Informationsflusses entlang des Produktentwicklungsprozesses. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 37, 1998 – ISBN 3-931466-36-1

Bd. 38 DEMEL, W.; SCHMITZ, G. (Hrsg.): Entwicklung und Transfer von Entwicklungssystemen der Mechatronik. Aachener Workshop TransMechatronik, 26. Juni 1998, Technologiezentrum am Europaplatz Aachen, HNI-Verlagsschriftenreihe, Paderborn, Band 38, 1998 – ISBN 3-931466-37-X

Bd. 39 GROBBEL, R.; LANGEMANN, T.: Leitfaden PPS-Systeme: Auswahl und Einführung in der Möbelindustrie. HNI-Verlagsschriftenreihe, Paderborn, Band 39, 1998 – ISBN 3-931466-38-8

Bd. 40 REHBEIN, P.: Tribologische Untersuchung von hochfrequent schwingenden Gleitkontakten für den Einsatz in Reibkraftschlüssigen Antrieben. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 40, 1998 – ISBN 3-931466-39-6

Bd. 41 DANGELMAIER, W.: KOMNET – Kommunikationsplattform für KMU-Netzwerke. HNI-Verlagsschriftenreihe, Paderborn, Band 41, 1998 – ISBN 3-931466-40-X

Bd. 42 KALLMEYER, F.: Eine Methode zur Modellierung prinzipieller Lösungen mechatronischer Systeme. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 42, 1998 – ISBN 3-931466-41-8

Bd. 43 TRAPP, R.: Stereoskopische Korrespondenzbestimmung mit impliziter Detektion von Okklusionen. Dissertation, Fachbereich für Elektrotechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 43, 1998 – ISBN 3-931466-42-6

Bd. 44 GAUSEMEIER, J.; FINK, A; SCHLAKE, O.: Grenzen überwinden - Zukünfte gestalten. 2. Paderborner Konferenz für Szenario-Management, HNI-Verlagsschriftenreihe, Paderborn, Band 44, 1998 – ISBN 3-931466-43-4

Bd. 45 wird noch vergeben!

Bd. 46 VÖCKING, B.: Static and Dynamic Data Management in Networks. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 46, 1998 – ISBN 3-931466-45-0

Bd. 47 SCHEKELMANN, A.: Materialflußsteuerung auf der Basis des Wissens mehrerer Experten. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 47, 1999 – ISBN 3-931466-46-9

Bd. 48 GECK-MÜGGE, K.: Herleitung und Spezifikation generischer Bausteine zur einheitlichen Modellierung von Fertigungsinformationen für die Fertigungssteuerung. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 48, 1999 – ISBN 3-931466-47-7

Bd. 49 WALLASCHEK, J.; LÜCKEL, J.; LITTMANN, W.: Heinz Nixdorf Symposium on Mechatronics and Advanced Motion Control. 3. Internationales Heinz Nixdorf Symposium, HNI-Verlagsschriftenreihe, Paderborn, Band 49, 1999 – ISBN 3-931466-48-5

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 50 FINK, A.: Szenariogestützte Führung industrieller Produktionsunternehmen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 50, 1999 – ISBN 3-931466-49-3

Bd. 51 HOLTKAMP, R.: Ein objektorientiertes Rahmenwerk zur Erstellung individueller, verteilter Fertigungslenkungssysteme. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 51, 1999 – ISBN 3-931466-50-7

Bd. 52 KUHN, A.: Referenzmodelle für Produktionsprozesse zur Untersuchung und Gestaltung von PPS-Aufgaben. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 52, 1999 – ISBN 3-931466-51-5

Bd. 53 SIEBE, A.: Systematik der Umsetzung von IT-orientierten Veränderungsprojekten in dynamischen Umfeldern. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 53, 1999 – ISBN 3-931466-52-3

Bd. 54 KLAHOLD, R. F.: Dimensionierung komplexer Produktionsnetzwerke. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 54, 1999 – ISBN 3-931466-53-1

Bd. 55 SCHÜRHOLZ, A.: Synthese eines Modells zur simulationsgestützten Potentialanalyse der Distribution. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 55, 1999 – ISBN 3-931466-54-X

Bd. 56 GEHNEN, G.: Integriertes Netzwerk zur Fertigungssteuerung und –automatisierung. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 56, 1999 – ISBN 3-931466-55-8

Bd. 57 KRESS, S.: Architektur eines workflowbasierten Planungsinstruments für die technische Auftragsbearbeitung unter besonderer Berücksichtigung des Einsatzes der Telearbeit. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 57, 1999 – ISBN 3-931466-56-6

Bd. 58 THIELEMANN, F.: Integrierte Methodik zur Gestaltung von Leistungserstellungsprozessen mittels Workflowmanagement. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 58, 1999 – ISBN 3-931466-57-4

Bd. 59 KROME, J.: Modelle zur Untersuchung des Schwingungsverhaltens von Statoren für piezoelektrische Ultraschall-Wanderwellen-Motoren. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 59, 1999 – ISBN 3-931466-58-2

Bd. 60 DEMEL, W.; SCHMITZ , G. (Hrsg.): Entwicklung und Transfer von Entwicklungssystemen der Mechatronik. Krefelder Workshop TransMechatronik, 24. August 1999 Fachhochschule Niederrhein, HNI-Verlagsschriftenreihe, Paderborn, Band 60, 1999 – ISBN 3-931466-59-0

Bd. 61 LANGEMANN, T.: Modellierung als Kernfunktion einer systemorientierten Analyse und Bewertung der diskreten Produktion. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 61, 1999 – ISBN 3-931466-60-4

Bd. 62 KÜMMEL, M.: Integration von Methoden und Werkzeugen zur Entwicklung von mechatronischen Systemen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 62, 1999 – ISBN 3-931466-61-2

Bd. 63 LUKOVSZKI, T.: New Results on Geometric Spanners and Their Applications. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 63, 1999 – ISBN 3-931466-62-0

Bd. 64 LÖFFLER, A.; MONDADA, F.; RÜCKERT, U. (Hrsg.): Experiments with the Mini-Robot Khepera, Proceedings of the 1st International Khepera Workshop. HNI-Verlagsschriftenreihe, Paderborn, Band 64, 1999 – ISBN 3-931466-63-9

Bd. 65 SCHÄFERMEIER, U.; BISCHOFF, C.: KMUnet - Ein Konzept zur ablauforganisatorischen Gestaltung der Lieferanteneinbindung. HNI-Verlagsschriftenreihe, Paderborn, Band 65, 2000 – ISBN 3-931466-64-7

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 66 HOLTHÖFER, N.: Regeln in einer Mengenplanung unter Ausbringungsgrenzen. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 66, 2000 – ISBN 3-931466-69-8

Bd. 67 SCHLAKE, O.: Verfahren zur kooperativen Szenario-Erstellung in Industrieunternehmen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 67, 2000 – ISBN 3-931466-66- 3

Bd. 68 LEWANDOWSKI, A.: Methode zur Gestaltung von Leistungserstellungsprozessen in Industrieunternehmen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 68, 2000 – ISBN 3-931466-67-1

Bd. 69 SCHMIDTMANN, A.: Eine Spezifikationssprache für die Fertigungslenkung. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 69, 2000 – ISBN 3-931466-68-X

Bd. 70 GROBBEL, R.: Eine Referenzarchitektur für Kooperationsbörsen. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 70, 2000 – ISBN 3-931466-69-8

Bd. 71 WESSEL, R.: Modelocked Waveguide Lasers in Lithium Niobate. Dissertation, Fachbereich für Physik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 71, 2000 – ISBN 3-931466-70-1

Bd. 72 LÖFFLER, A.: Energetische Modellierung neuronaler Signalverarbeitung. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 72, 2000 – ISBN 3-931433-71-X

Bd. 73 LUDWIG, L. A.: Computational Intelligence in der Produktionswirtschaft. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 73, 2000 – ISBN 3-931466-72-8

Bd. 74 WENSKI, R.: Eine objektorientierte Systemkomponente zur Workflow-Modellierung und -Ausführung unter besonderer Berücksichtigung der Telekooperation. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 74, 2000 – ISBN 3-931466-73-6

Bd. 75 GRASMANN, M.: Produktkonfiguration auf Basis von Engineering Data Management-Systemen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 75, 2000 – ISBN 3-931466-74-4

Bd. 76 DITZE, C.: Towards Operating System Synthesis. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 76, 2000 – ISBN 3-931466-75-2

Bd. 77 KÖRNER, T.: Analog VLSI Implementation of a Local Cluster Neural Network. Dissertation, Fachbereich für Elektrotechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 77, 2000 – ISBN 3-931466-76-0

Bd. 78 SCHEIDELER, C.: Probabilistic Methods for Coordination Problems. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 78, 2000 – ISBN 3-931466-77-9

Bd. 79 GAUSEMEIER, J.; LINDEMANN, U.; REINHART, G.; WIENDAHL, H.-P.: Kooperatives Produktengineering - Ein neues Selbstverständnis des ingenieurmäßigen Wirkens. HNI-Verlagsschriftenreihe, Paderborn, Band 79, 2000 – ISBN 3-931466-78-7

Bd. 80 GAUSEMEIER, J.; LÜCKEL, J.: Entwicklungsumgebungen Mechatronik - Methoden und Werkzeuge zur Entwicklung mechatronischer Systeme. HNI-Verlagsschriftenreihe, Paderborn, Band 80, 2000 – ISBN 3-931466-79-5

Bd. 81 RIEPING, I.: Communication in Parallel Systems-Models, Algorithms and Implementations. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 81, 2000 – ISBN 3-931466-80-9

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 82 GAUSEMEIER, J; LÜCKEL, J.: Auf dem Weg zu den Produkten für die Märkte von morgen. 4. Internationales Heinz Nixdorf Symposium, HNI-Verlagsschriftenreihe, Paderborn, Band 82, 2000 – ISBN 3-931466-81-7

Bd. 83 DEL CASTILLO, G.: The ASM Workbench - A Tool Environment for Computer-Aided Analysis and Validation of Abstract State Machine Models. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 83, 2000 – ISBN 3-931466-82-5

Bd. 84 SCHÄFERMEIER, U.: Eine Methode zur systemorientierten organisatorischen Gestaltung der Zweckaufgabenverrichtung in kooperativen Verbünden; Klassifikation, Aufgabenzuordnung. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 84, 2000 – ISBN 3-931466-83-3

Bd. 85 KRÜGER, J.: Ganzheitliche Beherrschung von Abläufen in und zwischen soziotechnischen Systemen: Ein Beitrag zur Modellbildung und zum paradigmatischen Verständnis von Industrieunternehmen zur Integration von Mensch und Maschine; Prozess und Struktur. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 85, 2000 – ISBN 3-931466-84-1

Bd. 86 BARTSCHER, T.: Methoden des Integrierten Workflowmanagements (IWFM). Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 86, 2000 – ISBN 3-931466-85-X

Bd. 87 QUINTANILLA, J.: Ein Verifikationsansatz für eine netzbasierte Modellierungsmethode für Fertigungssteuerungssysteme. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 87, 2000 – ISBN 3-931466-86-8

Bd. 88 PREIS, R.: Analyses and Design of Efficient Graph Partitioning Methods. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 88, 2001 – ISBN 3-931466-87-6

Bd. 89 wird noch vergeben!

Bd. 90 WESTERMANN, M.: Caching in Networks: Non-Uniform Algorithms and Memory Capacity Constraints. Dissertation, Fachbereich für Informatik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 90, 2001 – ISBN 3-931466-89-2

Bd. 91 LEMKE, J.: Nutzenorientierte Planung des Einsatzes von CAD- / CAE-Systemen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 91, 2001 – ISBN 3-935433-00-X

Bd. 92 VON BOHUSZEWICZ, O.: Eine Methode zur Visualisierung von Geschäftsprozessen in einer virtuellen Umgebung. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 92, 2001 – ISBN 3-935433-01-8

Bd. 93 BÖRNCHEN, T.: Zur Entwicklung dynamischer Komponenten für variables Kraftfahrzeug-Scheinwerferlicht. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 93, 2001 – ISBN 3-935433-02-6

Bd. 94 WINDELER, I.: Auswahl von Restrukturierungsprojekten in Forschungs- und Entwicklungsorganisationen der Automobilindustrie. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 94, 2001 – ISBN 3-935433-03-4

Bd. 95 WOLFF, C.: Parallele Simulation großer pulscodierter neuronaler Netze. Dissertation, Fachbereich für Elektrotechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 95, 2001 – ISBN 3-935433-04-2

Bd. 96 HENKE, A.: Modellierung, Simulation und Optimierung piezoelektrischer Stellsysteme. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 96, 2001 – ISBN 3-935433-05-0

Bd. 97 RÜCKERT, U.; SITTE, J.; WITKOWSKI, U. (Hrsg.): Autonomous Minirobots for Research and Edutainment AMiRE2001. 5. Internationales Heinz Nixdorf Symposium, HNI-Verlagsschriftenreihe, Paderborn, Band 97, 2001 – ISBN 3-935433-06-9

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 98 LI, P.: Datenkonversion für den Datenaustausch in verteilten Fertigungs-Lenkungssystemen. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 98, 2001 – ISBN 9-935433-07-7

Bd. 99 BRANDT, C.: Eine modellbasierte Methode zum strukturierten Entwurf virtueller Umgebungen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 99, 2001 – ISBN 9-935433-08-5

Bd. 100 WLEKLINSKI, C.: Methode zur Effektivitäts- und Effizienzbewertung der Entwicklung maschinenbaulicher Anlagen. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 100, 2001 – ISBN-3-935433-09-3

Bd. 101 HEMSEL, T.: Untersuchung und Weiterentwicklung linearer piezoelektrischer Schwingungsantriebe. Dissertation, Fachbereich für Maschinentechnik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 101, 2001 – ISBN 3-935433-10-7

Bd. 102 MAUERMANN, H.: Leitfaden zur Erhöhung der Logistikqualität durch Analyse und Neugestaltung der Versorgungsketten. Dissertation, Fachbereich für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 102, 2001 – ISBN 3-935433-11-5

Bd. 103 WAGENBLAßT, D.: Eine Analysemethode zur Beurteilung der Funktionssicherheit von gemischt analog-digitalen Schaltungen. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 103, 2002 – ISBN 3-935433-12-3

Bd. 104 PORRMANN, M.: Leistungsbewertung eingebetteter Neurocomputersysteme. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 104, 2002 – ISBN 3-935433-13-1

Bd. 105 SEIFERT, L.: Methodik zum Aufbau von Informationsmodellen für Electronic Business in der Produktentwicklung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 105, 2002 – ISBN 3-935433-14-X

Bd. 106 SOETEBEER, M.: Methode zur Modellierung, Kontrolle und Steuerung von Produktstrategien. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 106, 2002 – ISBN 3-935433-15-8

Bd. 107 GAUSEMEIER, J.; GRAFE, M. (Hrsg.): 1. Paderborner Workshop Augmented & Virtual Reality in der Produktentstehung. HNI-Verlagsschriftenreihe, Paderborn, Band 107, 2002 – ISBN 3-935433-16-6

Bd. 108 FLATH, M.: Methode zur Konzipierung mechatronischer Produkte. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 108, 2002 – ISBN 3-935433-17-4

Bd. 109 AVENARIUS, J.: Methoden zur Suche und Informationsbereitstellung von Lösungselementen für die Entwicklung mechatronischer Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 109, 2002 – ISBN 3-935433-18-2

Bd. 110 HELMKE, S.: Eine simulationsgegestützte Methode für Budgetentscheidungen im Kundenbindungsmanagement. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 110, 2002 – ISBN 3-935433-19-0

Bd. 111 CZUBAYKO, R.: Rechnerinterne Repräsentation von informationsverarbeitenden Lösungselementen für die verteilte kooperative Produktentwicklung in der Mechatronik. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 111, 2002 – ISBN 3-935433-20-4

Bd. 112 GOLDSCHMIDT, S.: Anwendung mengenorientierter numerischer Methoden zur Analyse nichtlinearer dynamischer Systeme am Beispiel der Spurführungsdynamik von Schienenfahrzeugen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 112, 2002 – ISBN 3-935433-21-2

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 113 LEHMANN, T.: Towards Device Driver Synthesis. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 113, 2002 – ISBN 3-935433-22-0

Bd. 114 HÄRTEL, W.: Issueorientierte Frühaufklärung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 114, 2002 – ISBN 3-935433-23-9

Bd. 115 ZIEGLER, M.: Zur Berechenbarkeit reeller geometrischer Probleme. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 115, 2002 – ISBN 3-935433-24-7

Bd. 116 SCHMIDT, M.: Neuronale Assoziativspeicher im Information Retrieval. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 116, 2003 – ISBN 3-935433-25-5

Bd. 117 EL-KEBBE, D. A.: Towards the MaSHReC Manufacturing System under Real-Time Constraints. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 117, 2003 – ISBN 3-935433-26-3

Bd. 118 PUSCH, R.: Personalplanung und -entwicklung in einem integrierten Vorgehensmodell zur Einführung von PDM-Systemen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 118, 2003 – ISBN 3-935433-27-1

Bd. 119 SOHLER, C.: Property Testing and Geometry. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 119, 2003 – ISBN 3-935433-28-X

Bd. 120 KESPOHL, H.: Dynamisches Matching – Ein agentenbasiertes Verfahren zur Unterstützung des Kooperativen Produktengineering durch Wissens- und Technologietransfer. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 120, 2003 – ISBN 3-935433-29-8

Bd. 121 MOLT, T.: Eine domänenübergreifende Softwarespezifikationstechnik für automatisierte Fertigungsanlagen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 121, 2003 – ISBN 3-935433-30-1

Bd. 122 GAUSEMEIER, J.; LÜCKEL, J.; WALLASCHEK, J. (Hrsg.): 1. Paderborner Workshop Intelligente mechatronische Systeme. HNI-Verlagsschriftenreihe, Paderborn, Band 122, 2003 – ISBN 3-935433-31-X

Bd. 123 GAUSEMEIER, J.; GRAFE, M. (Hrsg.): 2. Paderborner Workshop Augmented & Virtual Reality in der Produktentstehung. HNI-Verlagsschriftenreihe, Paderborn, Band 123, 2003 – ISBN 3-935433-32-8

Bd. 124 LITTMANN, W.: Piezoelektrische resonant betriebene Ultraschall-Leistungswandler mit nichtlinearen mechanischen Randbedingungen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 124, 2003 – ISBN 3-935433-33-6

Bd. 125 WICKORD, W.: Zur Anwendung probabilistischer Methoden in den frühen Phasen des Systementwurfs. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 125, 2003 – ISBN 3-935433-34-4

Bd. 126 HEITTMANN, A.: Ressourceneffiziente Architekturen neuronaler Assoziativspeicher. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 126, 2003 – ISBN 3-935433-35-2

Bd. 127 WITKOWSKI, U.: Einbettung selbstorganisierender Karten in autonome Miniroboter. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 127, 2003 – ISBN 3-935433-36-0

Bd. 128 BOBDA, C.: Synthesis of Dataflow Graphs for Reconfigurable Systems using Temporal Partitioning and Temporal Placement. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 128, 2003 – ISBN 3-935433-37-9

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 129 HELLER, F.: Wissensbasiertes Online-Störungsmanagement flexibler, hoch automatisierter Montagesysteme. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 129, 2003 – ISBN 3-935433-38-7

Bd. 130 KÜHN, A.: Systematik des Ideenmanagements im Produktentstehungsprozess. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 130, 2003 – ISBN 3-935433-39-5

Bd. 131 KEIL-SLAWIK, R.; BRENNECKE, A.; HOHENHAUS, M.: ISIS -Installationshandbuch für lernförderliche Infrastrukturen. HNI-Verlagsschriftenreihe, Paderborn, Band 131, 2003 – ISBN 3-935433-40-9

Bd. 132 OULD HAMADY, M.: Ein Ansatz zur Gestaltung des operativen Fertigungsmanagements innerhalb der Lieferkette. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 132, 2003 – ISBN 3-935433-41-7

Bd. 133 HOLTZ, C.: Theoretical Analysis of Unsupervised On-line Learning through Soft Competition. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 133, 2003 – ISBN 3-935433-42-5

Bd. 134 UEBEL, M.: Ein Modell zur Steuerung der Kundenbearbeitung im Rahmen des Vertriebsmanagements. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 134, 2003 – ISBN 3-935433-43-3

Bd. 135 BRINKMANN, A.: Verteilte Algorithmen zur Datenplazierung und zum Routing in gegnerischen Netzwerken. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 135, 2003 – ISBN 3-935433-44-1

Bd. 136 FRÜND, E.: Aktive Kompensation von periodischen Schwingungen an rotierenden Walzen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 136, 2003 – ISBN 3-935433-45-X

Bd. 137 KEIL-SLAWIK, R. (Hrsg.): Digitale Medien in der Hochschule: Infrastrukturen im Wandel. HNI-Verlagsschriftenreihe, Paderborn, Band 137, 2004 – ISBN 3-935433-46-8

Bd. 138 STORCK, H.: Optimierung der Kontaktvorgänge bei Wanderwellenmotoren. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 138, 2004 – ISBN 3-935433-47-6

Bd. 139 KALTE, H.: Einbettung dynamisch rekonfigurierbarer Hardwarearchitekturen in eine Universalprozessorumgebung. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 139, 2004 – ISBN 3-935433-48-4

Bd. 140 ISKE, B.: Modellierung und effiziente Nutzung aktiver Infrarotsensorik in autonomen Systemen. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 140, 2004 – ISBN 3-935433-49-2

Bd. 141 BÄTZEL, D.: Methode zur Ermittlung und Bewertung von Strategiealternativen im Kontext Fertigungstechnik. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 141, 2004 – ISBN 3-935433-50-6

Bd. 142 BÖKE, C.: Automatic Configuration of Real-Time Operating Systems and Real-Time Communication Systems for Distributed Embedded Applications. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 142, 2004 – ISBN 3-935433-51-4

Bd. 143 KÖCKERLING, M.: Methodische Entwicklung und Optimierung der Wirkstruktur mechatronischer Produkte. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 143, 2004 – ISBN 3-935433-52-2

Bd. 144 HENZLER, S: Methodik zur Konzeption der Struktur und der Regelung leistungsverzweigter Getriebe mit Toroidvariator. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 144, 2004 – ISBN 3-935433-53-0

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 145 GAUSEMEIER, J.; LÜCKEL, J.; WALLASCHEK, J. (Hrsg.): 2. Paderborner Workshop Intelligente mechatronische Systeme. HNI-Verlagsschriftenreihe, Paderborn, Band 145, 2004 – ISBN 3-935433-54-9

Bd. 146 LESSING, H.: Prozess zur multivariaten Prognose von Produktionsprogrammen für eine effiziente Kapazitätsplanung bei typisierten Dienstleistungen. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 146, 2004 – ISBN 3-935433-55-7

Bd. 147 HAMOUDIA, H.: Planerische Ablaufgestaltung bei prozessorientierten Dienstleistungen. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 147, 2004 – ISBN 3-935433-56-5

Bd. 148 BUSCH, A.: Kollaborative Änderungsplanung in Unternehmensnetzwerken der Serienfertigung – ein verhandlungsbasierter Ansatz zur interorganisationalen Koordination bei Störungen. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 148, 2004 – ISBN 3-935433-57-3

Bd. 149 GAUSEMEIER, J.; GRAFE, M. (Hrsg.): 3. Paderborner Workshop Augmented & Virtual Reality in der Produktentstehung. HNI-Verlagsschriftenreihe, Paderborn, Band 149, 2004 – ISBN 3-935433-58-1

Bd.150 MEYER, B.: Value-Adding Logistics for a World Assembly Line. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 150, 2004 – ISBN 3-935433-59-X

Bd. 151 GRIENITZ, V.: Methodik zur Erstellung von Technologieszenarien für die strategische Technologieplanung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 151, 2004 – ISBN 3-9354 33-60-3

Bd. 152 FRANKE, H.: Eine Methode zur unternehmensübergreifenden Transportdisposition durch synchron und asynchron kommunizierende Agenten. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 152, 2004 – ISBN 3-9354 33-61-1

Bd. 153 SALZWEDEL, K. A.: Data Distribution Algorithms for Storage Networks. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 153, 2004 – ISBN 3-935433-62-X

Bd. 154 RÄCKE, H.: Data Management and Routing in General Networks. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 154, 2004 – ISBN 3-935433-63-8

Bd. 155 FRANK, U.; GIESE, H.; KLEIN, F.; OBERSCHELP, O.; SCHMIDT, A.; SCHULZ, B.; VÖCKING, H.; WITTING, K.; GAUSEMEIER, J. (Hrsg.): Selbstoptimierende Systeme des Maschinenbaus – Definitionen und Konzepte. HNI-Verlagsschriftenreihe, Paderborn, Band 155, 2004 – ISBN 3-935433-64-6

Bd. 156 MÖHRINGER, S.: Entwicklungsmethodik für mechatronische Systeme. Habilitation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 156, 2004 – ISBN 3-935433-65-4

Bd. 157 FAHRENTHOLZ, M.: Konzeption eines Betriebskonzepts für ein bedarfsgesteuertes schienengebundenes Shuttle-System. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 157, 2004 – ISBN 3-935433-66-2

Bd. 158 GAJEWSKI, T.: Referenzmodell zur Beschreibung der Geschäftsprozesse von After-Sales-Dienstleistungen unter besonderer Berücksichtigung des Mobile Business. Dissertation Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 158, 2004 – ISBN 3-935433-67-0

Bd. 159 RÜTHER, M.: Ein Beitrag zur klassifizierenden Modularisierung von Verfahren für die Produktionsplanung. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 159, 2004 – ISBN 3-935433-68-9

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 160 MUECK, B.: Eine Methode zur benutzerstimulierten detaillierungsvarianten Berechnung von diskreten Simulationen von Materialflüssen. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 160, 2004 – ISBN 3-935433-69-7

Bd. 161 LANGEN, D.: Abschätzung des Ressourcenbedarfs von hochintegrierten mikroelektronischen Systemen. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 161, 2005 – ISBN 3-935433-70-0

Bd. 162 ORLIK, L.: Wissensbasierte Entscheidungshilfe für die strategische Produktplanung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 162, 2005 – ISBN 3-935433-71-9

Bd. 163 GAUSEMEIER, J.; RAMMIG, F.; SCHÄFER, W.; WALLASCHEK, J. (Hrsg.): 3. Paderborner Workshop Intelligente mechatronische Systeme. HNI-Verlagsschriftenreihe, Paderborn, Band 163, 2005 – ISBN 3-935433-72-7

Bd. 164 FISCHER, M.: Design, Analysis, and Evaluation of a Data Structure for Distributed Virtual Environments. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 164, 2005 – ISBN 3-935433-73-5

Bd. 165 MATYSCZOK, C.: Dynamische Kantenextraktion - Ein Verfahren zur Generierung von Tracking-Informationen für Augmented Reality-Anwendungen auf Basis von 3D-Referenzmodellen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 165, 2005 – ISBN 3-935433-74-3

Bd. 166 JANIA, T.: Änderungsmanagement auf Basis eines integrierten Prozess- und Produktdatenmodells mit dem Ziel einer durchgängigen Komplexitätsbewertung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 166, 2005 – ISBN 3-935433-75-1

Bd. 167 GAUSEMEIER, J.; GRAFE, M. (Hrsg.): 4. Paderborner Workshop Augmented & Virtual Reality in der Produktentstehung. HNI-Verlagsschriftenreihe, Paderborn, Band 167, 2005 – ISBN 3-935433-76-X

Bd. 168 VOLBERT, K.: Geometric Spanners for Topology Control in Wireless Networks. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 168, 2005 – ISBN 3-935433-77-8

Bd. 169 ROSLAK, J.: Entwicklung eines aktiven Scheinwerfersystems zur blendungsfreien Ausleuchtung des Verkehrsraumes. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 167, 2005 – ISBN 3-935433-78-6

Bd. 170 EMMRICH, A.: Ein Beitrag zur systematischen Entwicklung produktorientierter Dienstleistungen. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 170, 2005 – ISBN 3-935433-79-4

Bd. 171 NOWACZYK, O.: Explorationen: Ein Ansatz zur Entwicklung hochgradig interaktiver Lernbausteine. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 171, 2005 – ISBN 3-935433-80-8

Bd. 172 MAHMOUD, K.: Theoretical and experimental investigations on a new adaptive duo servo drum brake with high and constant brake shoe factor. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 172, 2005 – ISBN 3-935433-81-6

Bd. 173 KLIEWER, G.: Optimierung in der Flugplanung: Netzwerkentwurf und Flottenzuweisung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 173, 2005 – ISBN 3-935433-82-4

Bd. 174 BALÁŽOVÁ, M.: Methode zur Leistungsbewertung und Leistungssteigerung der Mechatronikentwicklung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 174, 2005 – ISBN 3-935433-83-2

Bd. 175 FRANK, U.: Spezifikationstechnik zur Beschreibung der Prinziplösung selbstoptimierender Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 175, 2005 – ISBN 3-935433-84-0

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 176 BERGER, T.: Methode zur Entwicklung und Bewertung innovativer Technologiestrategien. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 176, 2005 – ISBN 3-935433-85-9

Bd. 177 BERSSENBRÜGGE, J.: Virtual Nightdrive - Ein Verfahren zur Darstellung der komplexen Lichtverteilungen moderner Scheinwerfersysteme im Rahmen einer virtuellen Nachtfahrt. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 177, 2005 – ISBN 3-935433-86-7

Bd. 178 GAUSEMEIER, J. (Hrsg.): Vorausschau und Technologieplanung. 1. Symposium für Vorausschau und Technologieplanung Heinz Nixdorf Institut, 3. und 4. November 2005, Schloß Neuhardenberg, HNI-Verlagsschriftenreihe, Paderborn, Band 178, 2005 – ISBN 3-935433-87-5

Bd. 179 FU, B.: Piezoelectric actuator design via multiobjective optimization methods. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 179, 2005 – ISBN 3-935433-88-3

Bd. 180 WALLASCHEK, J.; HEMSEL, T.; MRACEK, M.: Proceedings of the 2nd International Workshop on Piezoelectric Materials and Applications in Actuators. HNI-Verlagsschriftenreihe, Paderborn, Band 180, 2005 – ISBN 3-935433-89-1

Bd. 181 MEYER AUF DER HEIDE, F.; MONIEN, B. (Hrsg.): New Trends in Parallel & Distributed Computing. 6. Internationales Heinz Nixdorf Symposium, 17. und 18. Januar 2006, HNI-Verlagsschriftenreihe, Paderborn, Band 181, 2006 – ISBN 3-939350-00-1

Bd. 182 HEIDENREICH, J.: Adaptierbare Änderungsplanung der Mengen und Kapazitäten in Produktionsnetzwerken der Serienfertigung. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 182, 2006 – ISBN 3-939350-01-X

Bd. 183 PAPE, U.: Umsetzung eines SCM-Konzeptes zum Liefermanagement in Liefernetzwerken der Serienfertigung. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 183, 2006 – ISBN 3-939350-02-8

Bd. 184 BINGER, V.: Konzeption eines wissensbasierten Instruments für die strategische Vorausschau im Kontext der Szenariotechnik. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 184, 2006 – ISBN 3-939350-03-6

Bd. 185 KRIESEL, C.: Szenarioorientierte Unternehmensstrukturoptimierung – Strategische Standort- und Produktionsplanung. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 185, 2006 – ISBN 3-939350-04-4

Bd. 186 KLEIN, J.: Efficient collision detection for point and polygon based models. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 186, 2006 – ISBN 3-939350-05-2

Bd. 187 WORTMANN, R.: Methodische Entwicklung von Echtzeit 3D-Anwendungen für Schulung und Präsentation. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 187, 2006 – ISBN 3-939350-06-0

Bd. 188 GAUSEMEIER, J.; GRAFE, M. (Hrsg.): 5. Paderborner Workshop Augmented & Virtual Reality in der Produktentstehung. HNI-Verlagsschriftenreihe, Paderborn, Band 188, 2006 – ISBN 3-939350-07-9

Bd. 189 GAUSEMEIER, J.; RAMMIG, F.; SCHÄFER, W.; TRÄCHTLER, A.; WALLASCHEK, J. (Hrsg.): 4. Paderborner Workshop Entwurf mechatronischer Systeme. HNI-Verlagsschriftenreihe, Paderborn, Band 189, 2006 – ISBN 3-939350-08-7

Bd. 190 DAMEROW, V.: Average and Smoothed Complexity of Geometric Structures. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 190, 2006 – ISBN 3-939350-09-5

Bd. 191 GIESE, H.; NIGGEMANN, O. (Hrsg.): Postworkshop Proceedings of the 3rd Workshop on Object-oriented Modeling of Embedded Real-Time Systems (OMER 3), HNI-Verlagsschriftenreihe, Paderborn, Band 191, 2006 – ISBN 3-939350-10-9

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 192 RADKOWSKI, R.: Anwendung evolutionärer Algorithmen zur Unterstützung des Entwurfs selbstoptimierender Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 192, 2006 – ISBN 3-939350-11-7

Bd. 193 SHEN, Q.: A Method for Composing Virtual Prototypes of Mechatronic Systems in Virtual Environments. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 193, 2006 – ISBN 3-939350-12-5

Bd. 194 REDENIUS, A.: Verfahren zur Planung von Entwicklungsprozessen für fortgeschrittene mechatronische Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 194, 2006 – ISBN 3-939350-13-3

Bd. 195 KUHL, P.: Anpassung der Lichtverteilung des Abblendlichtes an den vertikalen Straßenverlauf. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 195, 2006 – ISBN 3-939350-14-1

Bd. 196 MICHELS, J. S.: Integrative Spezifikation von Produkt- und Produktionssystemkonzeptionen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 196, 2006 – ISBN 3-939350-15-X

Bd. 197 RIPS, S.: Adaptive Steuerung der Lastverteilung datenparalleler Anwendungen in Grid-Umgebungen. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 197, 2006 – ISBN 3-939350-16-8

Bd. 198 GAUSEMEIER, J. (Hrsg.): Vorausschau und Technologieplanung. 2. Symposium für Vorausschau und Technologieplanung Heinz Nixdorf Institut, 9. und 10. November 2006, Schloß Neuhardenberg, HNI-Verlagsschriftenreihe, Paderborn, Band 198, 2006 – ISBN 3-939350-17-6

Bd. 199 FRANKE, W.: Wiederverwendungsorientierte Herleitung von Inter-Fachkomponentenkonzepten für Lagerverwaltungssoftwaresysteme. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 199, 2006 – ISBN 978-3-939350-18-7

Bd. 200 SCHEIDELER, P.: Ein Beitrag zur erfahrungsbasierten Selbstoptimierung einer Menge technisch homogener fahrerloser Fahrzeuge. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 200, 2006 – ISBN 978-3-939350-19-4

Bd. 201 KÖSTERS, C.: Ein ontologiebasiertes Modell zur Beschreibung der Abläufe in einem Produktionssystem unter besonderer Berücksichtigung einer diskreten Produktion. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 201, 2006 – ISBN 978-3-939350-20-0

Bd. 202 HALFMEIER, S.: Modellierung und Regelung von Halbtoroidvariationen in leistungsverzweigten Getriebestrukturen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 202, 2006 – ISBN 978-3-939350-21-7

Bd. 203 RÜHRUP, S.: Position-based Routing Strategies. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 203, 2006 – ISBN 978-3-939350-22-4

Bd. 204 SCHMIDT, A.: Wirkmuster zur Selbstoptimierung – Konstrukte für den Entwurf selbstoptimierender Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 204, 2006 – ISBN 978-3-939350-23-1

Bd. 205 IHMOR, S.: Modeling and Automated Synthesis of Reconfigurable Interfaces. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 205, 2006 – ISBN 978-3-939350-24-8

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 206 ECKES, R.: Augmented Reality – basiertes Verfahren zur Unterstützung des Anlaufprozesses von automatisierten Fertigungssystemen. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 206, 2007 – ISBN 978-3-939350-25-5

Bd. 207 STEFFEN, D.: Ein Verfahren zur Produktstrukturierung für fortgeschrittene mechatronische Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 207, 2007 – ISBN 978-3-939350-26-2

Bd. 208 LAROQUE, C.: Ein mehrbenutzerfähiges Werkzeug zur Modellierung und richtungsoffenen Simulation von wahlweise objekt- und funktionsorientiert gegliederten Fertigungssystemen. Dissertation, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 208, 2007 – ISBN 978-3-939350-27-9

Bd. 209 GAUSEMEIER, J.; GRAFE, M. (Hrsg.): 6. Paderborner Workshop Augmented & Virtual Reality in der Produktentstehung. HNI-Verlagsschriftenreihe, Paderborn, Band 209, 2007 – ISBN 978-3-939350-28-6

Bd. 210 GAUSEMEIER, J.; RAMMIG, F.; SCHÄFER, W.; TRÄCHTLER, A.; WALLASCHEK, J. (Hrsg.): 5. Paderborner Workshop Entwurf mechatronischer Systeme. HNI-Verlagsschriftenreihe, Paderborn, Band 210, 2007 – ISBN 978-3-939350-29-3

Bd. 211 KAUSCHKE, R.: Systematik zur lichttechnischen Gestaltung von aktiven Scheinwerfern. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 211, 2007 – ISBN 978-3-939350-30-9

Bd. 212 DU, J.: Zellen-basierte Dienst-Entdeckung für Roboternetzwerke. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 212, 2007 – ISBN 978-3-939350-31-6

Bd. 213 DANNE, K.: Real-Time Multitasking in Embedded Systems Based on Reconfigurable Hardware. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 213, 2007 – ISBN 978-3-939350-32-3

Bd. 214 EICKHOFF, R.: Fehlertolerante neuronale Netze zur Approximation von Funktionen. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 214, 2007 – ISBN 978-3-939350-33-0

Bd. 215 KÖSTER, M.: Analyse und Entwurf von Methoden zur Ressourcenverwaltung partiell rekonfigurierbarer Architekturen. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 215, 2007 – ISBN 978-3-939350-34-7

Bd. 216 RÜCKERT, U.; SITTE, J.; WITKOWSKI, U.: Proceedings of the 4th International Symposium on Autonomous Minirobots for Research and Edutainment – AMiRE2007. Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 216, 2007 – ISBN 978-3-939350-35-4

Bd. 217 PHAM VAN, T.: Proactive Ad Hoc Devices for Relaying Real-Time Video Packets. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 217, 2007 – ISBN 978-3-939350-36-1

Bd. 218 VIENENKÖTTER, A.: Methodik zur Entwicklung von Innovations- und Technologie-Roadmaps. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 218, 2007 – ISBN 978-3-939350-37-8

Bd. 219 GAUSEMEIER, J. (Hrsg.): Vorausschau und Technologieplanung. 3. Symposium für Vorausschau und Technologieplanung Heinz Nixdorf Institut, 29. und 30. November 2007, Miele & Cie. KG Gütersloh, HNI-Verlagsschriftenreihe, Paderborn, Band 219, 2007 – ISBN 978-3-939350-38-5

Bd. 220 FRÜND, J.: Eine Architekurkonzeption für eine skalierbare mobile Augmented Reality Anwendung für die Produktpräsentation. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 220, 2007 – ISBN 978-3-939350-39-2

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn

Bände der HNI-Verlagsschriftenreihe

Bd. 221 PEITZ, T.: Methodik zur Produktoptimierung mechanisch elektronischer Baugruppen durch die Technologie MID (Molded Interconnect Devices). Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 221, 2007 – ISBN 978-3-939350-40-8

Bd. 222 MEYER AUF DER HEIDE, F. (Hrsg.): The European Integrated Project "Dynamically Evolving, Large Scale Information Systems (DELIS)", Proceedings of the Final Workshop, Barcelona, February 27-28, 2008, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 222, 2008 – ISBN 978-3-939350-41-5

Bd. 223 GAUSEMEIER, J.; RAMMIG, F.; SCHÄFER, W. (Hrsg.): Self-optimizing Mechatronic Systems: Design the Future. 7. Internationales Heinz Nixdorf Symposium, 20. und 21. Februar 2008, HNI-Verlagsschriftenreihe, Paderborn, Band 223, 2008 – ISBN 978-3-939350-42-2

Bd. 224 RATH, M.: Methode zur Entwicklung hybrider Technologie- und Innovationsstrategien – am Beispiel des Automobils. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 224, 2008 – ISBN 978-3-939350-43-9

Bd. 225 GRÜNEWALD, M.: Protokollverarbeitung mit integrierten Multiprozessoren in drahtlosen Ad-hoc-Netzwerken. Dissertation, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 225, 2008 – ISBN 978-3-939350-44-6

Bd. 226 STRAUSS, S.: Theoretische und experimentelle Untersuchungen zum Einsatz gepulster Halbleiterlichtquellen in der Kraftfahrzeugbeleuchtung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 226, 2008 – ISBN 978-3-939350-45-3

Bd. 227 ZEIDLER, C.: Systematik der Materialflussplanung in der frühen Phase der Produktionssystementwicklung. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Paderborn, Band 227, 2008 – ISBN 978-3-939350-46-0

Bezugsadresse:
Heinz Nixdorf Institut
Universität Paderborn
Fürstenallee 11
33102 Paderborn