Kapitel 9

# ZUSAMMENFASSUNG UND AUSBLICK

## 9.1 Rückblick

Autonome mobile Systeme sind physische Agenten, die sich in dynamischen und unvorhersagbaren Umgebungen befinden. Der klassische Ansatz zur Realisierung von Autonomie basiert auf logischen Grundlagen. Die zentrale These dabei ist, daß intelligentes Verhalten durch eine Schlußfolgerungskomponente, die auf einem symbolischen Weltmodell operiert, realisiert werden kann. Man spricht in diesem Zusammenhang auch von Planung. Obwohl diese Systeme innerhalb einiger Anwendungsgebiete durchaus beachtenswerte Leistungen erzielen konnten, haben ihre Anwendungen in autonomen Systemen, die in komplexen und dynamischen realen Umgebungen agieren, nur unzufriedenstellende Ergebnisse erbracht. Die wenigen Systeme, die konstruiert wurden, weisen viele Unzulänglichkeiten wie mangelnde Robustheit und lange Antwortzeiten auf.

Als Alternative zu den klassischen Systemen wurden in den letzten Jahren verhaltensbasierte Systeme vorgeschlagen. Das Hauptaugenmerk in diesen Systemen liegt auf der engen Kopplung zwischen Perzeption (perception) und Aktion (action). Dies geschieht mit Hilfe einer Anzahl von speziellen reaktiven Prozessen, die meistens schnell und einfach sind, und die ein Minimum an expliziter Zustandsinformation enthalten. Das Ergebnis ist im allgemeinen eine hoch spezialisierte und angepaßte Lösung zu einer bestimmten Agent-Umwelt Situation. Diese Architekturen verlangen aber, daß der Designer des Systems in der Lage ist, die nötigen Prozesse zu entwerfen und für die richtigen Verbindungen zwischen ihnen zu sorgen. Mit steigender Komplexität heutiger Aufgaben wird es aber immer schwieriger, solchen Anforderungen zu genügen. Dieser Ansatz weist

außerdem, wegen der Festverdrahtung der Kontrollstruktur, einen großen Mangel an Flexibilität auf. Die Fähigkeiten des Agenten beschränken sich auf die Verhaltensmuster, die der Designer implementiert hat.

Im Rahmen dieser Arbeit wurde versucht, eine neue Steuerungsarchitektur für autonome mobile Systeme zu entwickeln, die die Vorteile beider oben erwähnten Ansätze nutzen soll.

Ein Modell, das die wichtigsten Eigenschaften autonomer Systeme, nämlich Reaktivität, Zielorientierung und Adaptivität berücksichtigt, wurde entworfen. Die Grundlage für dieses Modell ist eine Menge von konkurrierenden Prozessen, auch Verhaltensmuster oder Verhaltensmodule genannt, und eine auf Prioritäten basierende Arbitrierungsstrategie. Jedes Verhaltensmuster hat zu jedem Zeitpunkt und bezüglich jedes Ziels (goal: Die zu erledigende Aufgabe oder Task) eine bestimmte Priorität. Die Priorität spiegelt die Wichtigkeit des Verhaltensmusters für den Umgang mit der aktuellen Umweltsituation im Kontext des entsprechenden Ziels wider und wird dazu benutzt, das Verhaltensmuster zu bestimmen, das die Kontrolle im nächsten Kontrollzyklus übernehmen wird.

Die erste Frage, die sich daraus ergibt, ist: Wie können diese Prioritäten ermittelt werden? Ausgangspunkt ist das Anfangswissen des Designers über das System(z.B. Wissen über die Wirkung der Sensordaten auf die Verhaltensmuster) sowie eine Menge von Sensoren, die zu jedem Zeitpunkt Informationen über Umwelt und Ziel liefern. Eine der subtilen Eigenschaften der Sensorwerte ist, daß sie oft ungenau und manchmal auch falsch sind. Das Wissen des Designers ist meistens unvollständig.

Um ein technisches System zu bauen, das mit diesen Problemen fertig werden kann, müssen auch Konzepte zur Behandlung von Ungenauigkeit und Unvollständigkeit vorhanden sein. Ein solches Konzept ist z.B. Adaptivität. Das System ist adaptiv, wenn es in der Lage ist, sich selbst anzupassen und sein Verhalten zu verbessern. Adaptivität kann mit Hilfe von Techniken des maschinellen Lernens erreicht werden. Neuronale Netze sowie Methoden zum Lernen aus Erfahrung (Reinforcement Lernen) sind sehr nützliche Werkzeuge in diesem Zusammenhang. Neuronale Netze sind in der Lage on-line zu lernen und haben gute Generalisierungsfähigkeiten. Ungenaue und unvollständige Sensordaten können weiterhin eine Ausgabe erzeugen, die ursprünglich mit vollständigeren Daten as-

soziiert wurde. Reinforcement Lernen stellt einen Mechanismus dar, in dem der Designer Anforderungen und Zielvorgaben an das System allein durch Bewertung („Belohnung" (reward) und „Bestrafung" (punishment)) seiner Leistungsfähigkeit hinsichtlich der zu lösenden Aufgabe artikulieren kann. Einmal durchgeführte Aktionen werden somit als „Erfahrungen" im weiteren Betrieb nutzbringend eingesetzt.

In diesem Modell werden die Prioritäten mit Hilfe dieser Methoden gelernt. Zuerst wird versucht eine selbstorganisierende Karte (self-organizing map) zu konstruieren und gleichzeitig das Anfangswissen des Designers in diese Struktur zu integrieren. Damit wird das Problem der ungenauen und falschen Sensordaten weitgehend gelöst und gleichzeitig die Voraussetzung für die Anwendung von Reinforcement Lernen, nämlich die Generalisierung über Zustände, geschaffen. Wenn dies geschehen ist, wird Reinforcement Lernen angewandt, um die anfangs nicht optimale Steuerung zu verbessern und die Unvollständigkeit soweit wie möglich zu beheben. Dies geschieht mit Hilfe einer neuen Methode, die auf der Kombination von Exploration und Algorithmen der Dynamischen Programmierung beruht. Simulationsergebnisse haben die besondere Nützlichkeit dieser Vorgehensweise gezeigt. Mit einigen Grundfähigkeiten ausgestattet ist somit das System in der Lage, seine Funktionalität im Einsatz erfahrungsgestützt zu erweitern und zu verbessern und auch in unvorhersehbaren Situationen sinnvolles Verhalten zu zeigen.

Da nicht jedes beliebig komplexe Verhalten auf diese Weise leicht zu realisieren ist, stellt sich die Frage, wie der Agent eine komplexe Aufgabe schrittweise durch Koordination von bereits erlernten Verhaltensweisen erledigen kann. Zur Lösung dieses Problems wurde wiederum das Lernen aus Erfahrung benutzt. Bei der neuentwickelten Koordinationsmethode wird ein dynamisches neuronales Netz benutzt, um die Menge von Sequenzen von Parametervektorpaaren (ein Paar besteht aus einem Eingabeparametervektor und dem dazugehörigen Kontrollparametervektor) zu partitionieren. Der Agent versucht dabei, sich an erfolgreiche Sequenzen zu erinnern. In jedem Schritt wird die Sequenz, die am meisten Erfolg verspricht, benutzt. Abhängig vom Erfolg und der Vorhersagekraft der benutzten Sequenz wird dann die Sequenzmenge angepaßt. Die Zweckmäßigkeit dieser Methode wurde wiederum mit Hilfe von Simulationen erprobt.

Um „intelligente", lernfähige Systeme zu schaffen, die autonom in einer dynamischen Umwelt agieren können, müssen geeignete Vorstrukturierungen (a-priori-Wissen) einer-

seits und Lernalgorithmen andererseits in Lernarchitekturen vereinigt werden. Um dies zu leisten, ist es erforderlich, zu untersuchen, welche Arten von Information der menschliche Experte am einfachsten beitragen kann und welche Architekturen es am wirkungsvollsten erlauben, dieses Wissen mit Lernalgorithmen zu verknüpfen. Erste erfolgreiche Schritte in diese Richtung wurden in der vorliegenden Arbeit durch die Verbindung von verhaltensbasierten Ansätzen mit Techniken des maschinellen Lernens unternommen.

In den Experimenten, die in der vorliegenden Arbeit beschrieben sind, wurden simulierte Agenten benutzt. Obwohl es auf den ersten Blick scheint, daß für die Untersuchung von Robotern, die in realen Umgebungen agieren, Simulation ungeeignet ist, haben sich simulierte Umgebungen beim Testen von Designoptionen als sehr nützlich erwiesen. Die Ergebnisse der Simulation scheinen auch robust genug zu sein, um auf reale Umgebungen ohne große Schwierigkeiten übertragen zu werden (vorausgesetzt, daß die sensorischen und motorischen Fähigkeiten der Roboter denen ihrer simulierten Entsprechungen ähnlich sind). Außerdem kann das Trainieren in einer simulierten Umgebung und die anschließende Übertragung des resultierenden Kontrollers auf einen realen Roboter eine interessante, alternative Designmethode sein, weil die Verwendung von realen Robotern sehr zeitaufwendig ist . Wenn diese Übertragung sich dennoch als schwierig oder unnütz erweist, dann sollte es mindestens in einigen Fällen möglich sein, Verhaltensmodule, die in simulierten Umgebungen erlernt wurden, als Ausgangspunkt für das Trainieren von realen Robotern zu benutzen.

## 9.2 Verbindung zu anderen Arbeiten

Die vorliegende Arbeit ist mit einer Reihe von neueren Forschungsarbeiten verwandt, die sich auf die Realisierung von kompletten, mit der physischen Welt eng verbundenen, künstlichen Agenten konzentrieren. Diese Systeme werden oft ,,eingebettete" (embedded) oder ,,situierte" (situated) Agenten genannt. Beispiele für diesen Trend stellen die in [1],[76], [23],[24], und [172] beschriebenen Arbeiten dar. Obwohl es wichtige Unterschiede zwischen den verschiedenen Ansätzen gibt, scheinen einige gemeinsame Punkte sich gut etabliert zu haben. Eine erste grundlegende Anforderung ist, daß Agenten in der Lage sein müssen, ihre Aktivität in der realen Welt und in Realzeit auszuüben. Ein anderer wichtiger Punkt ist, daß adaptives Verhalten nur aus der engen Kopplung des

Agenten mit seiner Umgebung entstehen kann. Die vorliegende Arbeit hat sich darauf konzentriert, solch eine Kopplung durch Benutzung von Selbstorganisation und Reinforcement Lernen zu erzielen. Die Kontrollstruktur wird während der Interaktion des Agenten mit seiner Umwelt dynamisch entwickelt.

Selbstorganisation ist eine Art von Verhalten, das dynamische Systeme aufweisen. Beispiele für dieses Verhalten sind in vielen Bereichen der Wissenschaft reichlich vorhanden (siehe z.B. [72]). Selbstorganisation verweist auf die Fähigkeit eines Systems, das unüberwachtes Lernen benutzt, spezifische Detektoren für verschiedene Signalmuster zu entwickeln. Unüberwachtes Lernen (z.B. [81]) unterscheidet sich von überwachtem Lernen (z.B. [137]) in der Art der Information, die dem System zugeführt wird. Bei überwachtem Lernen muß ein ,,Lehrer" die Klasse, zu der jedes Trainingsbeispiel gehört, liefern. Bei unüberwachtem Lernen geht es hingegen darum, Regularitäten in den Trainingsbeispielen zu finden. Selbstorganisation in lernenden, adaptiven Kontrollern ist nützlich, weil es unwahrscheinlich ist, daß der Designer detailliertes Wissen darüber hat, wie die Umgebung mittels der Sensoren des Agenten charakterisiert werden kann und folglich es nicht einfach ist, die passenden Kontrollparameterwerte zu finden, die der Kontroller benutzen soll, wenn er spezifischen Umgebungen gegenübersteht. Ein System, das in der Lage ist, Umgebungssituationen autonom zu charakterisieren, kann nützliche Kontrollparameter mit den Situationen assoziieren und den Kontroller erfolgreich adaptieren.

Reinforcement Lernen (siehe z.B. [157]) wurde in den letzten Jahren in vielen verschiedenen algorithmischen Rahmen studiert. In [159] zum Beispiel haben Sutton, Barto und Williams die Benutzung von Reinforcement Lernen für die direkte adaptive Steuerung vorgeschlagen. Mahadevan and Connell ([89]) haben eine ,,Subsumption Architecture" ([19]) implementiert, in der die Verhaltensmodule mit Hilfe einer erweiterten Version des Q-Learning-Algorithmus ([168]) lernen. Methoden des Reinforcement Lernens kombinieren Methoden zur Anpassung von Aktionsauswahl mit Methoden zur Einschätzung der langfristigen Konsequenzen von Aktionen. Die Grundidee in Algorithmen des Reinforcement Lernens wie Q-Learning ist, eine Funktion $Q(.,.)$ von Zuständen und Aktionen abzuschätzen. Dabei ist $Q(x,a)$ der Erwartungswert der diskontierten Summe der zukünftigen Rewards, die dadurch erzeugt werden, daß Aktion $a$ in Zustand $s$ ausgeführt wird und danach nur optimale Aktionen ausgewählt werden.

## 9.3 Einschränkungen

Mit Hilfe der in dieser Arbeit vorgestellten Lernmethoden ist es möglich, Agenten zu entwerfen, die selbständig ihre Robustheit verbessern können. Die Agenten sind auch in der Lage, selbständig komplexe Tasks zu erlernen und bereits vorhandene Kompetenzen zu koordinieren, um komplexere Ziele zu lösen. Es wurde allerdings vorausgesetzt, daß eine konsistente und vollständige Menge von Basis-Verhaltensmustern bereits existiert, und daß es außerdem möglich ist, für die zu lösende Aufgabe eine Komplexitätshierarchie (Goal-Hierarchie) zu definieren. Für allgemeine Problemstellungen ist es aber schwierig, die Konsistenz und die Vollständigkeit der Mengen von Basis-Verhaltensmustern zu garantieren. Es ist auch schwierig, passende Komplexitätshierarchien zu definieren.

Eine weitere Einschränkung besteht in den vielen Parametern, die bei der Anwendung der Lernalgorithmen vom Designer „ von Hand" passend eingestellt werden müssen. Für manche Parameter können allgemeine Einstellungsregeln empirisch gefunden werden. Viele andere Parameter sind aber von den Einzelheiten der Aufgabenstellung abhängig und lassen sich nur sehr schwer im voraus bestimmen.

Ein weiteres Problem besteht, wie in allen nicht analytischen Lösungen, in der Schwierigkeit, die Korrektheit der Lösung zu beweisen. Die detaillierten Problemlösungsaktionen des Agenten können nur zur Laufzeit bestimmt werden (Emergence). Einzelnes Verhalten wird durch eine komplexe Wechselwirkung zwischen Agent und Umgebung geregelt. Die genaue „Trajektorie", der der Agent folgen wird, kann nur entdeckt werden, indem man den Agenten in seiner Umgebung laufen läßt. Das Verhalten des Agenten kann somit zur Designzeit nicht eindeutig festgestellt werden. Dies führt zu Einschränkungen bei der Anwendung in sicherheitskritischen Gebieten, in denen es wichtig ist, daß das System seine Spezifikation erfüllt.

All diese Probleme und Einschränkungen erfordern weitere Forschungsarbeit. Eine wichtige Forschungsrichtung wäre es, den Automatisierungsprozeß voranzutreiben, um die Anzahl der Designentscheidungen, die vom Designer getroffen werden müssen, zu reduzieren. Man könnte zum Beispiel versuchen, aus der allgemeinen Aufgabenstellung automatisch eine Komplexitätshierarchie zu erzeugen. Diese Art von High-Level-Planung würde die Autonomie des Agenten entscheidend verbessern und zur Lösung einiger der oben erwähnten Probleme beitragen (z.B. Korrektheitsgarantien). Man könnte auch ver-

suchen, mehr automatische Parametereinstellungsmechanismen in die Lernalgorithmen zu integrieren. Das Testen der Lernarchitektur in realistischeren Umgebungen sollte die Aspekte der Interaktion zwischen Agent und Umgebung deutlicher machen und Anlaß zu Verbesserungen geben. Die Anwendbarkeit der Architektur für andere Arten von Agenten, z.B. Software-Agent, Multi-Agenten-Systeme, usw., könnte auch untersucht werden.

Insgesamt glaube ich, daß die vorliegende Arbeit die Wichtigkeit von Lernen für das Erreichen eines zufriedenstellenden Adaptierungslevels zwischen einem künstlichen Agenten und seiner Umgebung zeigt. Es ist aber klar, daß noch viel Forschungsarbeit notwendig ist, um verstehen zu können, ob mit dem hier beschriebenen Ansatz eine mit dem adaptiven Verhalten eines lebenden Systems vergleichbare Komplexitätsstufe erreicht werden kann.

## 9.4 Die Zukunft

Robotik-Agenten gehören zur allgemeinen Klasse von intelligenten Agenten, die ein neues Paradigma für die Entwicklung von Softwareanwendungen darstellen. Intelligente Agenten sind gegenwärtig Gegenstand von intensivem Interesse seitens vieler Unterbereiche der Informatik (siehe z.B. [73]). Agenten werden in einer zunehmend breiten Vielzahl von Anwendungen benutzt, die von vergleichsweise kleinen Systemen wie Email-Filter zu großen, offenen, komplexen und kritischen Systemen wie Flugverkehrssteuerung reichen. Obwohl es auf den ersten Blick erscheint, daß solche extrem verschiedenen Arten von Systemen nur wenig gemeinsam haben könnten, ist der Begriff *Agent* eine Schlüsselabstraktion und ein grundsätzliches und wichtiges neues Werkzeug für die Konstruktion solcher Systeme. Grund dafür ist, daß die Metapher von autonomen Ploblemlösungsinstanzen eine intuitive und natürliche Art für die Konzeptualisierung mehrerer Probleme darstellt.

Im Moment herrscht noch eine rege Diskussion darüber, welche Eigenschaften Agenten im einzelnen aufweisen sollen, welche Anforderungen an sie gestellt werden müssen und wie eine allgemeingültige Definition des Begriffs „intelligenter Agent" lauten könnte. Robuste und effiziente Systeme erfordern aber auf jeden Fall methodische und rigorose

Designmethoden. Es gibt mehrere generelle Ansätze zum Design von Agenten, von denen der Ansatz der Selbstadaptation am geeignetsten erscheint. Dabei wird davon ausgegangen, daß ein Agent aus der Beobachtung seiner Umwelt - insbesondere der Beobachtung seines Benutzers - lernt und schrittweise immer nützlicher wird.

Die vielversprechende Idee des Agenten, und insbesondere die des lernenden Agenten, eröffnet einen weiten Horizont neuer, intelligenterer und vor allem flexiblerer Möglichkeiten, mit dem Informationsangebot der Zukunft umzugehen. Durch sie vollzieht die Wissenschaft des Informationsmanagements einen weiteren Schritt hin zur Verselbständigung von Software und Computern. Damit geschieht hier eine Entwicklung parallel zum Fortschritt der Programmiersprachen, die, angefangen von Lochkarten, hin zu immer „höheren" Befehlen strebten und noch immer streben. Welchem progressiv Denkenden leuchten nicht die Augen bei dem Gedanken, endlich mit einer Maschine in der „höchsten Sprache", der menschlichen Lautsprache kommunizieren zu können !?

Doch bis zur Verwirklichung dieser euphorischen Vision - so sie denn überhaupt jemals erreicht wird - ist es noch ein weiter Weg und eine große Anforderung an die Informatik, diesen Weg begehbar zu machen.