Intrinsisch motivierte Exploration sensomotorischer Zustandsräume

Diplomarbeit

Matthias Kubisch

—

Labor für Neurorobotik
Institut für Informatik

Betreuer: Dr. Manfred Hild
Gutachter: Prof. Dr. Hans-Dieter Burkhard
Prof. Dr.-Ing. Beate Meffert

Berlin, den 20. September 2010

Zusammenfassung

Aktives und selbständiges Lernen ist ein effektiver Mechanismus, mit dem sich Mensch und Tier optimal an die Gegebenheiten der Umwelt anpassen können. Getrieben durch Neugierde, erforscht das frühkindliche Individuum in einem langfristig angelegten Lernprozess die Möglichkeiten seines Körpers und die Eigenschaften seiner Umwelt. In der vorliegenden Arbeit wird untersucht, wie dieser Lernprozess als Informationsverarbeitung modelliert und auf einer Maschine implementiert werden kann. Dazu wird ein selbständig lernendes, primitives Individuum aufgebaut.

Das Ergebnis der Arbeit ist ein unüberwachter Lernalgorithmus, welcher auf unbegrenzte Dauer ausgelegt ist und dem Individuum durch aktive Handlungen die Zusammenhänge von Körper und Umwelt offenbart. Der Wissenserwerb ist dabei unabhängig von einer konkreten Lernaufgabe. Die Implementation erfolgt unter Verwendung künstlicher neuronaler Netze und kann in Echtzeit berechnet werden. Dabei wird untersucht, welche Auswirkungen es auf das Verhalten des Individuums hat, wenn dieses sich selbst für erfolgreiches Lernen belohnt. Das resultierende Verhalten wird unter Variation verschiedener Morphologien, darunter ein einfaches Robotersystem, beobachtet. Dabei zeigt das Individuum gerichtetes Verhalten und es können, in Abhängigkeit von der Morphologie, spezifische Verhaltensmuster beobachtet werden.

Danksagung

Hiermit möchte ich mich bei all denen bedanken, die zum Gelingen dieser Arbeit beigetragen haben. Besonderer Dank gilt dabei den Herren Christian Thiele und Christian Benckendorff für ihr Engagement beim Gegenlesen der Arbeit und für ihre inhaltlichen Anregungen. Ich möchte mich auch bei Julia Pajonk bedanken, die mir half die Allgemeinverständlichkeit dieser Arbeit wesentlich anzuheben. Weiterhin danke ich Marianne Wessel und Richard Lemke für ihr Korrekturvorschläge.

Zu guter Letzt möchte ich mich bei Maria Wander bedanken, die mich so herzlich umsorgte, dass ich mich – befreit von den kleinen nervigen Dingen des Alltags – ganz und gar auf diese Arbeit konzentrieren konnte.

Vorwort

Maria und ich sind zu Besuch bei Marianne und Stefan. Die beiden sind gerade erst mit ihrem einjährigen Sohn Justus nach Berlin gezogen und wir begutachten den fortgeschrittenen Stand ihrer neuen Wohnung in der Kaskelstraße. Marianne ist arbeiten, und so kommt es, dass Maria, Stefan und ich uns im Kinderzimmer niederlassen und erzählen. Justus ist schwer mit der Überprüfung der Gesetzmäßigkeiten zur Schwerkraft anhand farbiger Holzbausteine beschäftigt, als ich beschließe, mich dazu zu gesellen. Der Einjährige ist mir längst aus dem Sinn, da ich nun selbst mein Zeugnis in Architektur ablege und einen Turm erbaue, der in Höhe und Anmut seinesgleichen sucht. Er ist zudem perfekt in Statik und Symmetrie.

Justus erblickt ihn. Das Resultat seiner Reaktion ist wegen des Holzbodens nur schwer zu überhören. Der Troll zeigt sich über seine Leistung sichtlich begeistert und lacht mich ausgelassen an.

Inhaltsverzeichnis

1 Einleitung

Der Mensch entwickelt im Laufe seines Lebens Fähigkeiten, die er sich, wie viele Tiere auch, durch aktives und selbständiges Lernen aneignet. Besonders während der frühkindlichen Entwicklung sind dabei zwei Besonderheiten über die Artenvielfalt hinweg zu beobachten. Auffälligstes Merkmal ist die Tatsache, dass das kindliche Individuum beim selbstbestimmten Lernen offensichtlich so freudig erregt ist, dass es dabei alles andere um sich herum vergisst. Augenscheinlich ist besonders beim Menschen zu beobachten, dass die durch Eigeninitiative oder spielerisch erlernten Fähigkeiten dabei auffällig oft wiederholt werden. Das Ausführen der neu erworbenen Fähigkeit, sowie die Zuführung der dadurch erzeugten Sinneswahrnehmungen, haben eine fast magische Anziehungskraft auf das kindliche Individuum und motivieren es anscheinend zur ständigen Wiederholung.

Wie funktioniert das Lernen bei Mensch und Tier in diesen frühkindlichen Entwicklungsphasen? Kann man diesen Prozess als Informationsverarbeitung modellieren und in einer Rechenmaschine nachbilden? Wenn ja, bestünde dann nicht Grund zur Annahme, dass sich mithilfe eines solchen Modells beobachtbare Voraussagen über das biologische Vorbild machen ließen? Die Eigenschaft aktiv zur Lebenszeit lernen zu können, verschafft einer Spezies offenbar entscheidende evolutionäre Vorteile. Welchen Weg nahm die Evolution von der Vorgabe fester Verhaltensmuster zu anpassungsfähigem Verhalten? Wie könnten erste Zwischenschritte ausgesehen haben?

Kleinkinder haben einen massiven Wissensdurst, welcher sich zunächst durch intensives Begutachten, Berühren und Ausprobieren alltäglicher Gegenstände und später durch exzessives Nachfragen äußert. Dabei ist die Art und Weise des Wissenserwerbs für den Beobachter oft unvorhersehbar und scheinbar nicht zielgerichtet, weil mit häufigem Wechsel der Aufmerksamkeit verbunden. Welcher Mechanismus befähigt ein Individuum zu diesem selbständigen Lernen und was sind seine Bestandteile? Kann der vollständige sensomotorische Lernapparat als Zusammenspiel von funktionalen Einzelteilen verstanden werden? Welche Wechselwirkungen ergeben sich zwischen den möglichen Komponenten? Der erfolgreiche Lernprozess wird anscheinend vom Lernenden als Genugtuung oder Spaß empfunden. Ist es somit möglich, dass Lernfortschritt als eine Primärmotivation interpretiert werden kann? Könnte man also ein abstraktes Motivationssystem nachbilden, welches den Fortschritt des Lernens misst und das Individuum in Abhängigkeit davon intrinsisch belohnt? Ist diese Form der Motivation, unabhängig von anderen Primärmotivationen, überhaupt untersuchbar?

Bisher gibt es noch keine Roboter, welche zu einem selbstbestimmten Lernen in vergleichbarer Weise fähig wären. Einen solchen zu bauen, ist heutzutage eine der größten Herausforderungen für die Wissenschaft. Fragestellungen und diesbezügliche Untersuchungen fasst man derzeit unter dem Begriff *developmental robotics*, auf Deutsch etwa »Entwicklungsrobotik«, zusammen. Wie aber implementiert man solch ein aufgabenunspezifisches, autonomes Lernen auf einem Roboter?

Das Ziel dieser Arbeit ist die Modellierung, Implementation und anschließende Untersuchung eines basalen Lernverfahrens, welches in der Lage ist, in einfacher Weise autonom Wissen über Körper und Umwelt zu erwerben und dieses stets zu aktualisieren. Der Prozess des Lernens wird dabei auf unbegrenzte Zeit ausgelegt und hat kein klassisches Lernziel, wie beispielsweise eine Balancieraufgabe. Das Lernziel ist erfolgreiches Lernen selbst und somit aufgabenunspezifisch. Es soll ein vollständiger Algorithmus aufgebaut werden und im Ergebnis ein selbständig lernendes, primitives Individuum entstehen. Zentral ist dabei die Frage, wie exploratives Verhalten erzeugt werden kann und ob dieses durch eine rein intrinsische Motivation gezielt beeinflusst und sogar gefördert werden kann. Dabei soll sich das Individuum intern selbst belohnen, wenn es erfolgreich etwas gelernt hat.

Für die Untersuchung gilt es vorerst die Fragestellung so weit es geht zu reduzieren. Die Funktion des Algorithmus soll daher zunächst an simulierten, abstrakten Testszenarien und später an einem Robotersystem, mit wenigen Freiheitsgraden, untersucht werden. Selbst solche reduzierten Systeme, bestehend aus zwei Motoren und zwei Sensoren, bieten für die Fragestellung der Arbeit genügend Komplexität. Für den Vorgang des Lernens bleibt somit der Zustand des Gesamtsystems überschaubar. Das Individuum beginnt den Lernprozess ohne explizite Vorkenntnisse und wird für die Untersuchungen von allen Erfordernissen, welche dem Selbsterhalt dienen, freigestellt. Um sich vollständig auf das Lernen zu konzentrieren und Wechselwirkungen mit anderen Motivationen auszuschließen, wird dem Individuum dazu ein unbegrenzter Energievorrat und die Gewissheit zur Verfügung gestellt, dass selbst zugeführter Schaden jederzeit ausgeschlossen ist. Letzten Endes wird das resultierende Verhalten des Individuums unter Variation verschiedener Morphologien und wichtiger Systemparameter untersucht.

Basis für den Ansatz ist die Hypothese, dass das Gehirn als Rechenmaschine verstanden werden kann, welche ihre Eingaben in Form von Sinneswahrnehmungen empfängt, verarbeitet und in Form von Interaktion mit der Umwelt wieder ausgibt. Als Handwerkzeug dient dazu die Methode des Konnektionismus, d. h. für die Realisierung einzelner Algorithmusbestandteile kommen zum Großteil die Bausteine Neuronen und Synapsen zum Einsatz. Als Lernverfahren für die künstlichen neuronalen Netze werden dabei bewährte Methoden mit wenigen oder unkritisch einzustellenden Parametern bevorzugt. Sofern es möglich ist, wird dazu auch die üblicherweise manuelle Einstellung konstanter Parameter durch angemessene lokale Regelprozesse erledigt. Der gesamte Algorithmus soll dabei fähig sein, alle Berechnungen in Echtzeit durchzuführen, immer im Hinblick auf eine reale Roboterplattform. Daher werden alle vorgestellten Methoden auf ihren Rechenzeit- und Speicherbedarf untersucht.

Der Aufbau der Arbeit gestaltet sich wie folgt: Die nächsten beiden Kapitel stellen alle erforderlichen Grundlagen für den weiteren Verlauf der Arbeit dar. In Kapitel 2 werden die der Arbeit zugrundeliegenden Konzepte vorgestellt. Dabei werden, im Hinblick auf Systeme mit begrenzten Ressourcen, die Voraussetzungen für ein Lernen auf unbestimmte Dauer festgelegt. Außerdem werden die Prinzipien der Selbstorganisation und -regulation vorgestellt und erläutert, was unter intrinsischer Motivation zu verstehen ist. Kapitel 3 liefert das notwendige theoretische Rüstzeug, welches für das Verständnis der darauffolgenden Kapitel hilfreich ist. Hier wird das verwendete Neuronenmodell definiert, es werden übliche Netzarchitekturen erläutert und die für

diese Arbeit verwendeten neuronalen Lernverfahren vorgestellt. Dabei wird die Funktionsweise mit gezielten Experimenten und Abbildungen erläutert und auf alternative Verfahren hingewiesen.

Das Kapitel 4 beschreibt im Überblick das Modell des Individuums, definiert dabei den sensomotorischen Apparat und erklärt Schritt für Schritt den geschlossenen Zyklus aus Wahrnehmung, Verarbeitung und Handlung. Die drei daran anschließenden Kapitel untersuchen die dazu nötigen Komponenten im Detail, beginnend mit der Zustandsidentifikation in Kapitel 5, gefolgt von separaten Untersuchungen zur diskreten Differentiation in Kapitel 6 und abschließend mit der Bewertung, Auswahl und Ausübung motorischer Aktionen in Kapitel 7. Die Details der Implementation werden in Kapitel 8 erläutert. Hier werden die verwendete Testumgebung und die Experimentieranordnungen vorgestellt und die empirischen Ergebnisse der Untersuchung analysiert. Das letzte Kapitel schließt die Arbeit mit einer Zusammenfassung ab und gibt auf Grundlage der gewonnenen Erfahrungen einen Ausblick auf mögliche Erweiterungen und Verbesserungen.

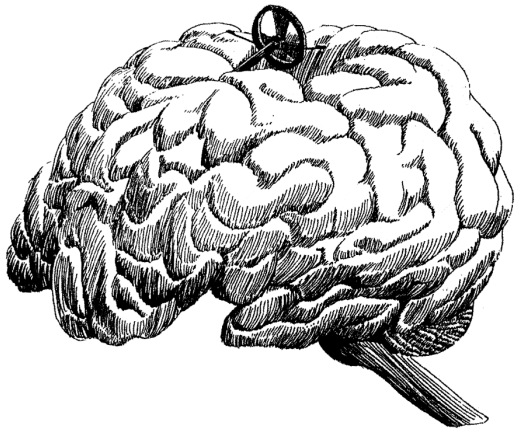

Abbildung 1.1: *Wer steuert?* von Hans-Georg Rauch. Abbildung entnommen aus [9].

2 Grundlegende Konzepte und Betrachtungsweisen

Dieses Kapitel legt den gedanklichen Hintergrund der Arbeit dar und erklärt die grundlegenden Konzepte. Der erste Abschnitt beschreibt, was im Zusammenhang mit dieser Arbeit unter »Körper« und »Lernen« zu verstehen ist. Der darauffolgende Abschnitt erläutert, welchen Anforderungen ein Lernvorgang auf Systemen mit begrenzten Ressourcen unterliegt und welche Voraussetzungen für ein Lernen auf unbegrenzte Dauer erfüllt sein müssen. Der dritte Abschnitt beschreibt die Prinzipien der Selbstregulation und -organisation und wie im Speziellen ein selbstorganisierendes Lernverfahren aus dem *Nichts* heraus eine Informationsstruktur aufbauen kann. Das Kapitel schließt mit der Einführung des Begriffs *Intrinsische Motivation*.

2.1 Körper, Umwelt und Lernen

Körper, Umwelt und Verhalten biologischer Individuen sind in besonderer Weise eng miteinander verflochten. Der Körperbau und das Verhalten sind in der Regel ausgezeichnet an die vorherrschenden Umweltbedingungen des jeweiligen Lebensraums angepasst. Diese Anpassung geschieht entweder über den Generationszyklus, d. h. durch Evolution, oder durch Lernen zur Lebenszeit. Das Einbeziehen des Körperbaus und der sensomotorischen Fähigkeiten in die Untersuchung maschineller Lernverfahren wird allgemein als *Embodiment* bezeichnet. Dazu dient die Annahme, dass für die Ausprägung eines angepassten Verhaltens ein Körper benötigt wird und dass die Interaktion mit der Umwelt eine notwendige Bedingung für den Erfolg der Anpassung ist [35]. Folglich muss sich ein Individuum sein Wissen über Körper und Umwelt durch aktive Bewegungen aneignen, d. h. es muss seine sensomotorischen Möglichkeiten in den frühen Lernstadien zuerst einmal *selbst explorieren*.

Eine konkrete Unterscheidung zwischen Körper und Umwelt ist bei der Betrachtung basaler Lernverfahren wenig dienlich. Um seine Umwelt zu verändern, verwendet ein Individuum im Allgemeinen dieselben Mechanismen, mit denen es auch seinen Körper beeinflusst. Anderseits verändert sich die Umwelt stetig selbst, oder wird durch andere verändert, was wiederum auf den Zustand des Individuums zurückwirken kann. Gelegentlich verschwindet die Grenze zwischen Körper und Umwelt sogar, wenn man deren grundlegende, physikalische Eigenschaften wie Druck und Temperatur betrachtet. Bezieht man zusätzlich auch noch Prothesen oder moderne mechatronische Körpergerätschaften mit ein, so ist die Unterscheidung gar ein philosophisches Problem.

Wenn die Umwelt sich in einer bestimmten Art und Weise verändert, aber das Individuum prinzipiell nicht in der Lage ist diese Veränderung mit seinen Sinnen wahrzunehmen, muss die Betrachtung dieser Eigenschaft konsequenterweise fallengelassen

werden. Unternimmt man stattdessen den Versuch, für die Untersuchungen die Umwelt von vornherein aus den Lernexperimenten auszuschließen und betrachtet *nur* den Körper, so verliert der Begriff *Körper* seine eigentliche Bedeutung und eine Unterscheidung ist wiederum hinfällig. Solange es also keine zwingenden Argumente für eine Unterscheidung gibt, und solange das Individuum mittels seiner Sinne nicht im Stande ist, verlässlich direkt oder indirekt einen solchen Unterschied auch wahrzunehmen, gibt es folglich keine Notwendigkeit eine Unterscheidung zwischen Körper und Umwelt künstlich aufrecht zu erhalten. Im Folgenden wird daher aus Sicht der Lernverfahren konsequent auf diese Unterscheidung verzichtet[1].

Lernen

Was bedeutet Lernen bei einem künstlichen Individuum? Dazu muss zunächst geklärt werden, was im Allgemeinen unter dem Begriff *Lernen* verstanden wird. In [51] steht dazu folgende Definition:

> »Unter Lernen versteht man den absichtlichen (intentionales Lernen) und den beiläufigen (inzidentelles und implizites Lernen), individuellen oder kollektiven Erwerb von geistigen, körperlichen, sozialen Kenntnissen, Fähigkeiten und Fertigkeiten. Aus lernpsychologischer Sicht wird Lernen als ein Prozess der relativ stabilen Veränderung des Verhaltens, Denkens oder Fühlens aufgrund von Erfahrung oder neu gewonnenen Einsichten und des Verständnisses (verarbeiteter Wahrnehmung der Umwelt oder Bewusstwerdung eigener Regungen) aufgefasst.«

Im Kontext dieser Arbeit wird unter dem Prozess des Lernens eine kontinuierliche Anpassung der *freien Parameter des Systems* zur Laufzeit verstanden. Diese Parameter sind z. B. die Stärken synaptischer Verbindungen, die mithilfe verschiedener Lernmechanismen verändert werden. Da es ausschließlich selbstregulative Mechanismen sind, wird hierbei unter Lernen ausdrücklich *kein Training* verstanden. Der Anspruch ist dabei, die Interaktion des Experimentators mit dem Lernenden auf ein Minimum zu reduzieren und dabei den ungestörten Ablauf der Experimente zu garantieren. Gerät der Lernende allerdings in Situationen, in denen das Lernen außerordentlich blockiert wird und aus denen er sich nicht selbst befreien kann, so sind Hilfestellungen zulässig. In Bezug auf diese Arbeit ist das vornehmliche Ziel, die körperlichen Fähigkeiten, d. h. den Umgang mit dem eigenen Körper, zu erproben. Dabei ist es nicht Teil der Arbeit, einen Mechanismus aufzubauen, welcher alle erprobten Fähigkeiten *dauerhaft erlernt*. Hier steht die gerichtete Selbstexploration im Vordergrund. Dazu ist allerdings eine funktionierende Wahrnehmung notwendig, welche gleichzeitig erlernt werden soll. Als Maß für den Erfolg des Lernprozesses dient also die Beobachtung einer »relativ stabilen Veränderung des Verhaltens« von anfänglich niedriger, zu kontinuierlich anwachsender Komplexität. Ebenso sollte ein zu Beginn zufälliges Verhalten, über die Zeit, eine für das Individuum spezifische Struktur aufweisen und in Abhängigkeit zum Körperbau und zu den sensomotorischen Fähigkeiten plausibel sein.

[1] Nichtsdestotrotz wird wahlweise der eine oder andere Begriff verwendet, da es kein präzises Wort gibt, welches beides gleichermaßen ausdrückt. Sie sind aber im Kontext dieser Arbeit als Synonyme zu verstehen.

Weiterhin heißt es in [51]:

> »Die Fähigkeit zu lernen ist für Mensch und Tier eine Grundvoraussetzung dafür, sich den Gegebenheiten des Lebens und der Umwelt anpassen zu können, darin sinnvoll zu agieren und sie gegebenenfalls im eigenen Interesse zu verändern. [...] Die Resultate des Lernprozesses sind nicht immer von den Lernenden in Worte fassbar (implizites Wissen) oder eindeutig messbar.«

Es besteht also nicht selten die Schwierigkeit die Resultate des Lernprozesses biologischer Individuen zu messen. Prinzipiell liegen in einem künstlichen System alle Vorgänge zur Analyse offen und es ist oft nur eine Frage der Lernarchitektur, wie gut sich die erlernten Strukturen analysieren lassen. Schwierig bleibt dennoch die Analyse des Verhaltens. Dabei mangelt es hauptsächlich an sinnvollen Vergleichsmöglichkeiten. Der Körperbau und die Lernmechanismen künstlicher Individuen sind entweder technisch bedingt zu einfach, um einem Vergleich mit der biologischen Vorbild standzuhalten, oder sind konzeptionell *bewusst reduziert*, um die Auswirkungen im Detail studieren zu können. Hierbei müssen also sinnvolle Maße gefunden werden, um die Lernresultate bezüglich des Verhaltens zu beschreiben.

2.2 Lernen unter beschränkter Rationalität

Biologische Individuen sammeln Informationen über Körper und Umwelt mit Hilfe ihre Sinnesorgane. Diese Wahrnehmungen sind aber prinzipiell nur ein kleiner Ausschnitt aller verfügbaren Informationen und somit steht für ausstehende Entscheidungen über zukünftige motorische Aktionen eben *nur ein individueller Blick* auf den *wirklichen Zustand der Welt* zur Verfügung. Angereichert mit Informationen aus der Erinnerung, wird das entstehende Bild präziser, aber es bleibt nach wie vor unvollständig. Um angemessen in einer allgemein hochdynamischen Umgebung zu reagieren, müssen solche Entscheidungen, welche also schon an und für sich auf unvollständigen Informationen beruhen, auch noch in kurzer und begrenzter Zeit getroffen werden. Diesen Umstand bezeichnet man als *beschränkte Rationalität*.

Die Rechenzeit und Speicherkapazität auf *Echtzeitsystemen*, wie autonomen mobilen Robotern, ist eine knappe Ressource. Dieser Tage werden in großer Zahl Mikrocontroller zum Auslesen der Sensoren und Ansteuern der Motoren von autonomen Robotern verwendet. Für das Ausrechnen von Lernverfahren werden bisweilen aber noch leistungsstarke Prozessoren eingesetzt, um als Hauptrecheneinheit zuständig für das Lernen, eine zentrale Rolle zu übernehmen. Je kleiner dabei die *Roboterplattform* wird, umso mehr stellt sich die Frage, ob zugunsten kleinerer und sparsamer Prozessoren, auf diese Hauptrecheneinheit auch verzichtet werden kann. Die geringere Leistungsaufnahme ist dabei erheblich von Vorteil, wenn die Stromversorgung ebenfalls mobil mitgeführt werden soll. Unabhängig davon welche Rechenarchitektur im Speziellen verwendet wird, lohnt ein kritischer Blick auf die Komplexität der verwendeten Algorithmen. Je gröber das System seine Berechnungen approximieren kann und dabei noch das Erwartete leistet, desto schlanker sind meist die verwendeten Algorithmen. Erwei-

tert man *Ockhams Rasiermesser*[2] um die Ausführungszeit und den Speicherverbrauch von Lernalgorithmen, so wäre von verschiedenen Verfahren mit gleichem Lernergebnis dasjenige zu bevorzugen, welches die geringsten Ressourcen verbraucht.

Beim Lernen auf Echtzeitsystemen werden sogenannte *Online*-Lernverfahren eingesetzt, um eine kontinuierliche Anpassung des Systems zur Laufzeit durchzuführen. Solche Verfahren sind darauf ausgelegt, in häufigen Aktualisierungsschritten behutsam die Parameter des Systems an die neue Situation anzupassen. Für diesen Aktualisierungszyklus steht dem System also nur ein kleines definiertes Zeitfenster zur Verfügung, um die Sensorik auszulesen, die Motorik anzusteuern und alle erforderlichen Berechnungen durchzuführen. Die Echtzeitbedingung fordert also ein, dass alle Berechnungen nach einer bestimmen Zeit definitiv abgeschlossen sind, oder andernfalls abgebrochen werden müssen. Ein entsprechendes Lernverfahren sollte im Idealfall so skalieren, dass die verfügbare Rechenzeit maximal ausgenutzt, aber nie überschritten wird. Das ist für diese Arbeit der Kern der beschränkten Rationalität.

Bezogen auf das lebenslange Lernen bedeutet diese Beschränkung der Rechenzeit- und Speicherressourcen, dass nicht beliebig lang Informationen im Sinne von *Erfahrung* angehäuft werden können. Ist die Kapazität erschöpft, so müssen Mechanismen wirksam werden, welche gezielt die gespeicherten Informationen absuchen und nach Möglichkeiten zur Rationalisierung fahnden. Im Wesentlichen bedeutet das, weniger wichtige Spezialisierungen aufzulösen und Ähnliches zusammenzufassen – also zu *generalisieren*. In Abschnitt 3.2.3 wird ein Verfahren untersucht und in Abschnitt 5.2 angepasst, welches ein vielversprechender Kandidat für diese Anforderungen ist. Des Weiteren wird bei den zur Auswahl stehenden Lernverfahren die Voraussetzung für lebenslanges Lernen und ein entsprechender Umgang mit beschränkter Rationalität gefordert. Konkret heißt das, dass die Verfahren per se *online*-fähig sein sollen und eine regelmäßige Nützlichkeitsprüfung der gesammelten Erfahrungen durchführen müssen. Das ist notwendig, um wieder Raum für neue Informationen zu schaffen, welche, aller Erwartung nach, einen höheren Nutzen haben werden. Solch ein Verfahren wird dann zwangsläufig mit dem Dilemma zwischen Stabilität und Plastizität konfrontiert und muss stetig zwischen dem Bewahren und Verwerfen von Informationen abwägen.

2.3 Selbstregulation und -organisation

Ein fundamentales Merkmal zur Unterteilung maschineller Lernverfahren ist *Überwachung*. Dahinter steckt im Wesentlichen nichts weiter als der Sachbestand, *ob* (und wie) Trainingsdaten zur Verfügung stehen. Ein Lernverfahren wird als überwacht bezeichnet, wenn für das Lernen vorbereitete Datenpaare aus Aufgabe und Lösung bereitstehen, sodass die eigene Lösung mit der Musterlösung verglichen und anhand dessen eine Korrektur vorgenommen werden kann. Im Gegensatz dazu bezeichnet man ein Lernverfahren als unüberwacht, wenn es keine Rückmeldung mit Korrekturvorschlägen bekommt. Das Lernverfahren hat dann meist ein abstraktes Lernziel, welches in den Lernmechanismus, der sogenannten *Lernregel* fest eingebaut ist. Häufig wird dazu ein *Wettbewerbslernen* generiert. Dieser Wettbewerb besteht zwischen mehreren ele-

[2] Ockhams Rasiermesser ist ein Sparsamkeitsprinzip und lautet: Von mehreren Theorien, die dasselbe erklären, ist diejenige zu bevorzugen, welche am *einfachsten* ist.

mentaren Lernmaschinen, welche dann kollektiv den eigentlichen Lernenden bilden. Beispielsweise geben dazu alle Elemente eine Schätzung über die richtige Lösung der gestellten Aufgabe ab. Das Element mit der besten Schätzung hat gewonnen und darf als Belohnung eine Anpassung seiner veränderlichen Lernparameter vornehmen. Ganz ohne eine Rückkopplung geht es also nicht, auch wenn diese schon in das Lernziel eingebaut ist. Im Allgemeinen findet alsbald eine Spezialisierung der einzelnen Elemente statt, wobei aus dem Wettbewerb eigentlich eher ein Teamspiel wird, indem jedes Element eine Nische besetzt. Die Organisationsstruktur gehört dabei genauso zum System wie seine Bewertung. Abschnitt 3.2.3 stellt ein derartiges Lernverfahren vor.

Emergenz

Das Interessante an unüberwachten Lernverfahren ist, dass sich aus dem Zusammenspiel vieler, an und für sich, einfacher Elemente ein komplexes Resultat abzeichnet. Oft kennt jedes Element nur seine eigenen Eigenschaften und die seiner unmittelbaren Nachbarschaft. Gelegentlich ist wenige, unspezifische, globale Information verfügbar, welche von allen Elementen gleichermaßen wahrgenommen werden kann. Entstehen in solchen Strukturen dann neue Eigenschaften, die das Einzelteil nicht besitzt, so bezeichnet man diese als *emergent*. Solch eine Emergenz kann also immer dort auftauchen, wo viele Einzelteile als Kollektiv zusammenwirken und dabei das Ganze mehr als die Summe seiner Teile ist, und dabei neuartige Eigenschaften hervorbringt.

Ein klassisches Beispiel einer emergenten Eigenschaft ist der Aggregatzustand eines Stoffes. Diese Eigenschaft tritt allerdings erst dann zutage, wenn eine gewisse Menge an Atomen oder Molekülen desselben Stoffes vorhanden sind. Ein einzelnes Wassermolekül ist demnach nicht nass. Der flüssige Zustand kann erst ab einer vergleichsweise großen Anzahl an Molekülen bestimmt werden und ist darüber hinaus durch den Bewegungszustand der Einzelteile bestimmt. Für die Beschreibung der Eigenschaft bildet die vorhandene mittlere kinetische Energie der Teilchen und ihre Geometrie die Ausgangsbasis für physikalische Modelle[3].

Ein weiteres Beispiel sind Schwärme. So besteht ein Schwarm meist aus vielen Lebewesen gleicher Art, wobei jedes Individuum sich an seinen unmittelbaren Nachbarn orientiert und sich daraus das Verhalten des ganzen Schwarms ergibt. Oft wird dabei die Bewegung nicht zwangsweise komplizierter, im Gegenteil, es ist sogar häufig eine Reduktion der Komplexität zu beobachten. In der Gesamtbewegung verhalten sich alle Individuen zusammen näherungsweise wie ein einziges Individuum. Dabei wird die Näherung mit wachsender Teilnehmerzahl meist besser. Dies ist ein Effekt, den man sich in der Physik zunutze macht, um einfache Modelle von (an sich) komplizierten Mehrteilchensystemen zu erstellen.

Bei emergenten Eigenschaften ist es nicht zwingend notwendig, dass alle Elemente in direktem Kontakt stehen und andauernd wechselwirken. Emergenz kann auch zeitlich verzögert auftreten. Betrachtet man beispielsweise die Entstehung von Trampelpfaden auf Wiesen und in Wäldern, so stellt man fest, dass ohne Wissen der einzelnen Individuen voneinander, und trotz zeitlich weit auseinander liegender Einzelaktionen, sich ein allein auf der Geometrie der Einzelteile und der Umgebung beruhender Effekt ergibt.

[3] Manche Physiker [26] gehen sogar von einer vollständig emergenten Struktur der Physik aus.

Das Wissen über derartige emergente Phänomene stellt in Bezug auf die Lernverfahren insofern eine neue Herausforderung dar. Man muss nun nach den lokalen Mechanismen der Einzelteile suchen, welche zu den emergenten Eigenschaften führen. Gesucht wird also nach lokalen Lernregeln, die im Startzustand noch keine erlernte Struktur aufweisen. Dabei bezeichnet das sogenannte *bootstrapping* die Eigenschaft des Systems, sich alle Informationen selbst zu generieren und somit Struktur aufzubauen. Das passiert wie folgt: Durch die eigenen Handlungen generiert man aktiv neue Wahrnehmungen. Diese werden von dem gerade selbst erst heranwachsenden, noch primitiven Wahrnehmungssystem verarbeitet und gespeichert, welches dann wiederum die Entscheidungsbasis für neue Aktionen ist. Dieser Kreislauf generiert kontinuierlich neue Informationen, woran sich das Wahrnehmungssystem selbständig anpassen kann. Mit der Zeit kann es die Implikationen der eigenen Aktionen immer besser vorhersehen. Somit wird auch die Bewertung der eigenen Aktionen präziser und es werden zunehmend komplexere Aktionen ausgewählt. Was in erster Linie nach einer Lügengeschichte des Baron Münchhausen[4] klingt, entfaltet sich, im Gegensatz dazu, zu einem emergenten Verfahren, dass in selbstorganisierter Form strukturiertes Verhalten aus dem Nichts aufbaut. Die Struktur in den Sensordaten liegt bereits vor, jedoch wird dem System darüber nichts mit auf den Weg gegeben. Es muss die Zusammenhänge selbst aufspüren. Dass dies hochgradig von der Wahrnehmungsarchitektur und dem Körperbau abhängig ist, liegt auf der Hand.

Homöostase

Die Selbstregulation (*Homöostase bzw. Homöodynamik*) ist ein fundamentales Naturprinzip biologischer Systeme. In unterschiedlichen Körperregionen der Organismen sind homöostatische Prozesse identifizierbar. Beispielsweise wird die Atemfrequenz des Menschen bei unterschiedlichen Belastungssituationen reguliert, um u. a. den Anforderungen des Sauerstoffbedarfs gerecht zu werden. Weitere Regelgrößen sind beispielsweise die Herzfrequenz und der Blutzuckerspiegel.

Dabei ist häufig eine Kaskadierung mehrerer Regelprozesse über verschiedene physikalische Größenordnungen beobachtbar. Ein anschauliches Beispiel dafür liefert die Reaktion des menschlichen Sehsystems auf abrupte Änderungen der Lichtintensität. An vorderster Front der Signalkette befindet sich der *Lidschlussreflex*, welcher bei starkem Lichteinfall – z. B. beim direkten Blick in die Sonne – das Augenlid sofort verschließt, um eine Schädigung der Netzhaut zu verhindern. Die nächste Stufe bildet der *Pupillenlichtreflex*, welcher ebenfalls versucht den Lichteinfall durch eine Verengung der Pupille zu reduzieren. Andererseits ist hier auch eine Pupillenerweiterung möglich, um den Lichteinfall ggf. erhöhen zu können. Hier handelt es sich also eigentlich um zwei Prozesse, welche im Wechselspiel interagieren. Die letzte Stufe der Regulation findet einige Größenordnungen darunter, und zwar direkt auf der Netzhaut statt. Dieser als *Hell- und Dunkeladaption* bekannte Prozess spielt sich unmittelbar in den Sehsinneszellen ab und regelt deren Sensitivität auf einfallende Photonen. Die Dunkeladaption der Sehsinneszellen läuft außerdem auf einer wesentlich längeren Zeitskala ab. Während der Lidschluss- und Pupillenlichtreflex im Bereich von wenigen Millisekunden

[4] Nach einer dieser Geschichten zog sich der Baron mitsamt seinem Pferd allein an seinem eigenen Schopf aus dem Sumpf.

passieren, kann die Dunkeladaption oft erst nach etwa 40 Minuten als abgeschlossen betrachtet werden.

Ein weiterer homöostatischer Prozess auf der Netzhaut ist die *chromatische Adaption*. Das Auge führt einen ständigen *automatischen* Weißabgleich durch, wodurch sich ein situationsspezifisches Empfinden der Farbtemperatur verliert. So ist es daher möglich, farbige Gegenstände in unterschiedlichen Beleuchtungssituationen als dieselbe Farbe besitzend zu identifizieren. Die chromatische Adaption ist im Wesentlichen kein weiterer Prozess. Sie kommt vielmehr dadurch zu Stande, dass die für unterschiedliche Wellenlängen des Lichts empfindlichen Sehsinneszellen jeweils ihren eigenen Adaptionsprozess durchlaufen.

Die meisten homöostatischen Prozesse lassen sich in einzelne, wechselwirkende Prozesse aufgliedern, welche durchaus auf unterschiedliche Anpassungsgeschwindigkeiten eingestellt seien können. So ist es nicht verwunderlich, dass die Reaktion auf eine Überreizung durch eine zu hohe Lichtintensität schlagartig ihre Wirkung hat (Schutzfunktion), während für die Anpassung an plötzliche Dunkelheit einige Sekunden bis Minuten verstreichen dürfen.

Eine Gemeinsamkeit homöostatischer Prozesse bei Sinnesorganen und -zellen ist neben der Schutzfunktion die Maximierung der Information. Ein konstanter Stimulus ist nach einiger Zeit relativ uninteressant und trägt, nach Shannon [41], kaum mehr Information. Hier kann es sinnvoll sein diesen konstanten Anteil im *Signal* langsam auszugleichen und dafür die Empfindlichkeit der Wahrnehmung zu erhöhen (Signal-Verstärkung), um somit nach bisher *verdeckten Informationen* im Signal zu suchen. Die Sensorinformation kann also fließend in ihrer Dynamik reduziert werden, was die Sensitivität auf subtile Information mit geringer Amplitude erhöht und relativ informationsarme oder gesättigte Signale vermeidet. Dabei ist es von Bedeutung, die zeitliche Reaktion der Regulation einige Größenordnungen über der Dynamik des eigentlichen Signals anzusetzen, um dem Signal nicht zusätzliche tieffrequente Anteile hinzuzufügen. In Abschnitt 3.2.1 wird ein derartiges Lernverfahren beschrieben, welches als einfaches Modell für homöostatische Prozesse auf der Ebene von Nervenzellen funktioniert.

2.4 Lernfortschritt als intrinsische Motivation

Sind alle grundlegenden physischen Bedürfnisse befriedigt, treten intrinsische Motive zutage. Sie generieren exploratives Verhalten, das neue Sinneswahrnehmungen produzieren oder die Art der Ausübung bestehender Fertigkeiten verbessert kann. Exploratives Verhalten wird als zentrale Grundlage erfolgreicher Lebensbewältigung angesehen. Explorative Individuen begeben sich in Situationen, mit denen sie noch nicht vertraut sind. Nun wird versucht, diese Situationen einzuordnen und sich in ihnen zu bewähren. Dabei werden neue Erfahrungen gemacht und dazugelernt. Je mehr unterschiedliche Situationen bereits in Erfahrung gebracht wurden, desto mehr Kontrolle hat das Individuum über neuartige Situationen, da es das Erlernte verallgemeinern und für die Bewältigung der neuen Situationen verwenden kann.

Darin ist ein evolutionärer Vorteil für das Individuum zu sehen. Eine aktive Exploration erzeugt zusätzliches Wissen über Körper und Umgebung, welches genutzt werden kann, um beispielsweise neue Nahrungsquellen aufzuspüren. Bei Lebewesen,

deren Verhalten durch Evolution im Wesentlichen *neuronal vorkodiert* ist, muss die Anpassung an veränderliche Umweltsituationen über den Generationszyklus erfolgen. Eine Anpassung durch Lernen funktioniert wesentlich schneller, da sie bereits zur Lebenszeit passiert und wird weiter beschleunigt, indem das Wissen an die Nachkommen weitergegeben wird. Dazu muss das Individuum einen ausgeprägten Mechanismus zum *Modelllernen*[5] besitzen, d. h. in der Lage sein, Verhaltensweisen durch Beobachtung und Imitation anderer Individuen zu erwerben, oder ggf. zu vermeiden. Die Anpassung kann also nur noch besser werden, wenn es zusätzlich Neugier entwickelt, um sich aktiv neue Sinneswahrnehmungen zu generieren und damit das Wissen über den eigenen Körper und die Umwelt zu aktualisieren und zu erweitern.

In [32, 31, 23] wird untersucht, in welcher Weise intrinsische Motivation modellierbar ist. Eine vielversprechende Annahme ist es, den erlangten Lernfortschritt des Individuums als intern vergebene Belohnung zu interpretieren. Dazu müsste das Individuum in adäquater Weise seinen Lernfortschritt messen und ihn unmittelbar in Beziehung zur aktuellen sensomotorischen Situation setzen. Es sollte nun vermehrt die Situationen aufsuchen, in denen es erfolgreich lernen konnte. Intrinsiche Motivation und exploratives Verhalten sind somit die Grundbausteine für ein aufgabenunspezifisches Lernen. Die Exploration erzeugt, durch den Zufall getrieben, neue Situationen und Sinnesreize. Die intrinsische Motivation gibt dabei die Richtung vor. Es ist erforderlich, dass die generierte Sinneswahrnehmung, d. h. die sensorische Information, gut zu dem bisherigen Zustand des lernenden Systems passt. Zu triviale Information hat keinen Mehrwert und ist ggf. zwecklos. Zu komplexe sensorische Information kann von der im Aufbau befindlichen Struktur möglicherweise nicht angemessen verarbeitet werden.

Die Unterscheidung von intrinsischer und extrinsischer Motivation ist nicht immer konsistent und wird, je nach Fachrichtung, mit anderen Vokabeln belegt. Intrinsische Motivation ist kein Synonym für *interne Motivation* [31]. Die Unterscheidung intern/extern soll nur aussagen *wo der Ursprung* für die Motivation liegt – innerhalb oder außerhalb des Individuums. Diese Unterscheidung ist aber nicht immer ganz schlüssig anwendbar bzw. verschieden interpretierbar und soll daher hier vermieden werden. Eine Unterscheidung nach intrinsisch bzw. extrinsisch sagt nichts über die Herkunft, sondern vielmehr über die Art der zugrundeliegenden Belohnung aus. Die intrinsische Motivation basiert auf dem Interesse oder dem Spaß an der Handlung selbst. Das zugehörige Bedürfnis ist folglich der Drang nach neuem Wissen, also die Neugier. Zu den extrinsischen Motiven werden folglich diejenigen gezählt, für welche die Handlung nur Mittel zum Zweck ist und welche nicht um ihrer selbst Willen ausgeführt werden. Die grundlegenden physischen Motive wie Hunger und Durst zu stillen, sowie die Vermeidung von Schmerz und Tod, sind demnach extrinsische Motive, wenngleich auch sie innerhalb des Körpers generiert werden. Im Zuge dieser Arbeit wird nur in den zwei genannten Kategorien unterschieden. Davon abgeleitete Motive werden hierfür nicht betrachtet.

In der vorliegenden Arbeit wird der Versuch unternommen, extrinsische Motive weitestgehend aus der Betrachtung auszuklammern. Dazu müssen Annahmen gemacht werden, um diesen Zustand für den jeweilige Morphologie sicherzustellen. Nun kann

[5] Lernen am Modell ist Lernen durch Beobachtung von Vorbildern und nicht mit dem Lernen eines internen Modells zu verwechseln.

man ein primitives, künstliches Individuum von seinen grundlegenden Überlebenstrieben befreien, indem man beispielsweise ein virtuelles Individuum erschafft und dabei auf eine simulierte physikalische Umgebung zurückgreift. Betrachtet man reale Systeme, wie beispielsweise einen kleinen Roboter, so müsste man eine kontinuierliche Stromversorgung bereitstellen und die Umwelt und den Körperbau so gestalten, dass selbst zugeführter Schaden vermieden wird. Für die meisten Roboterplattformen kann dieser Zustand hergestellt werden, indem man das zu verwendende Drehmoment der Motoren begrenzt oder alle Körperteile angemessen auspolstert. Zusätzlich kann man über regelmäßige Abkühlungspausen für die Motoren nachdenken, falls sich diese durch den Dauereinsatz merklich erwärmen.

Nachdem nun der konzeptionelle Hintergrund der Arbeit beschrieben wurde, folgt die Einführung des theoretischen Handwerkzeugs. Eine Methode die erdachten Prozesse zu formalisieren entstammt dabei direkt den identifizierten Strukturen im Gehirn. Es ist der Ansatz des *Konnektionismus* die Verarbeitung von Informationen als einen kollektiven Prozess vieler elementarer Einheiten, den Nervenzellen oder Neuronen, zu verstehen. Dabei ergeben sich vielfältige Möglichkeiten der Verschaltung mithilfe gewichteter Verbindungen, den sogenannten Synapsen. Das folgende Kapitel stellt diese Methodik vor und bildet somit den zweiten Teil der Grundlagen, auf denen diese Arbeit fußt.

3 Theoretisches Handwerkzeug und Grundlagen neuronaler Lernverfahren

Neuronale Netze sind ein universell einsetzbares Werkzeug zur Informationsverarbeitung. Mit wenigen Bestandteilen beschreiben sie einen vollständigen Bausatz für vielfältige Anwendungen. Im Rahmen dieser Arbeit kann nur ein kleiner Ausschnitt aus der Fülle an Verfahren vorgestellt werden. Dabei werden explizit nur die für diese Arbeit relevanten Aspekte behandelt und ggf. detaillierter beschrieben. Unter anderem wird die Funktionsweise durch gezielte Experimente erläutert. Das Kapitel ist wie folgt aufgebaut: Im ersten Abschnitt wird schrittweise das für die Arbeit verwendete Neuronenmodell aufgebaut und eine Übersicht über verschiedene Netzstrukturen gegeben. Der zweite Abschnitt stellt drei grundverschiedene neuronale Lernverfahren vor. Das erste ist eine homöostatische Lernregel für ein einzelnes Neuron. Darauf folgt die Herleitung eines bewährten Verfahrens für das Training mehrschichtiger Netze, wenn konkrete vorgegebene Daten erlernt werden sollen. Das Kapitel schließt mit der Vorstellung eines Vertreters der selbstorganisierenden Netzwerke.

3.1 Aufbau und Struktur künstlicher neuronaler Netze

Die Bestandteile künstlicher neuronaler Netze (KNN) sind im Wesentlichen Neuronen und Gewichte. Wird bei natürlichen Neuronen noch zwischen Axon, Dendrit und Synapse unterschieden so besitzt das abstrakte Modell oft nur noch eine gewichtete und gerichtete Verbindung zwischen zwei Neuronen, die allgemein als Synapse oder auch als Gewicht bezeichnet wird. Anschaulich kann man sich ein künstliches neuronales Netz als einen Graphen mit Knoten (Neuronen) und gerichteten Kanten (Gewichten) vorstellen. Beim Reproduzieren im Computer allerdings wechselt die Sichtweise zur Vektoralgebra. Betrachtet man die Werte aller n Eingänge $x_i \in \mathbb{R}$ mit $i = 1..n$ für ein Neuron als eine zusammengehörige Einheit, so lassen sie sich zu einem Vektor $\mathbf{x}$ zusammenfassen. Ebenfalls einen Vektor bilden die Gewichte $w_{ji} \in \mathbb{R}$, welche jeweils den Eingang i mit dem Neuron j verbinden. Hat ein Gewicht den Wert Null, so besteht keine Verbindung. Die Eingänge können dabei Netzeingaben (z. B. Sensordaten) oder die Ausgänge anderer Neuronen sein. In der Literatur sind zahlreiche Neuronenmodelle vorgestellt worden. Einen guten Überblick über etablierte und häufig verwendete Modelle gibt [19].

3.1.1 Definition des Neuronenmodells

Die Art und Weise, wie verschiedene Eingangssignale in ein Neuron gelangen, wird hier, wie auch in [19], als *effektiver Eingang* bezeichnet. Das in dieser Arbeit eingesetzte Neuronenmodell verwendet dazu das Skalarprodukt aus dem Gewichtsvektor und dem

Eingangsvektor. Der effektive Eingang eines Neurons j ist daher definiert als

$$a_j = \mathbf{w}_j{}^T \mathbf{x} = \sum_{i=1}^{n} w_{ji} x_i \tag{3.1}$$

und ist nichts weiter als die Summe der mit w_{ji} gewichteten Eingänge x_i. Diese Summe wird u. a. umso höher, je mehr *Ähnlichkeit* zwischen Eingangsvektor und Gewichtsvektor besteht. Ein Neuron wird somit stärker *aktiviert*, wenn seine Parameter zu den Eingaben passen. Dies wird noch deutlicher, wenn man sich eine alternative Darstellung des Skalarprodukts als

$$\mathbf{w}_j{}^T \mathbf{x} = w_j x \cos \measuredangle (\mathbf{w}_j, \mathbf{x})$$

ansieht, wobei $w_j = \|\mathbf{w}_j\|$ und $x = \|\mathbf{x}\|$ die Beträge, d. h. die Längen der Vektoren sind. Der effektive Eingang ist somit proportional zu den Längen der Vektoren und zu dem Kosinus des Winkels zwischen ihnen. Stehen der Eingangsvektor und der Gewichtsvektor senkrecht aufeinander, d. h. sind sie orthogonal, so ist der Kosinus gleich Null und das Skalarprodukt verschwindet. Sind sie hingegen parallel oder antiparallel ist der Betrag des Skalarprodukts am größten. Damit ist ein Neuron schon in der Lage als einfacher Musterdetektor zu funktionieren. Passt das an den Eingängen anliegende Signal zur eigenen Gewichtskonfiguration so ist der effektive Eingang hoch.

Ausgangsfunktion

Nachdem der effektive Eingang gebildet wurde, wird in der Regel eine *Ausgangsfunktion*[1] angewendet. Die Ausgangsfunktion in dem hier verwendeten Neuronenmodell ist der *Tangens Hyperbolicus* (tanh). Dieser ist streng monoton wachsend auf dem Definitionsbereich $(-\infty, +\infty)$ und in der Darstellung durch die Exponentialfunktion

$$y = f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

lässt sich gut als Wertebereich das offene Intervall $(-1, +1)$ ablesen, weil jeweils für $\lim_{x \to \pm\infty}$ die Exponentialterme je nach Vorzeichen des Exponenten wahlweise gegen 0 oder ∞ gehen. Somit liefert diese Ausgangsfunktion eine für die Nachbildung natürlicher Prozesse wichtige Eigenschaft: die *Sättigung*. Zu große Eingangssignale werden abgeschwächt und auf das Intervall $(-1, +1)$ beschränkt (vgl. dazu Abbildung 3.1). Liegt durch hohe Gewichte eine *Verstärkung* des Signals vor verhält sich der Tangens Hyperbolicus zunehmend wie die Signum-Funktion

$$\operatorname{sgn}(x) := \begin{cases} +1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases},$$

[1] In der Literatur herrscht an dieser Stelle ein regelrechtes Begriffs-Wirrwarr. Für diese Arbeit wird der seltenere Terminus *Ausgangsfunktion* [19] verwendet, u. a. um die Abgrenzung zur Übertragungs- bzw. Transferfunktion der Filtertheorie zu erhalten.

womit dieses Neuronenmodell auch für Anwendungen mit hoch gesättigten Signalen (z. B. für die Nachbildung binärer Logik-Gatter) verwendbar ist. Dabei bleibt die Ausgangsfunktion überall stetig differenzierbar. Darüber hinaus ist der Tangens Hyperbolicus $f \in C^\infty$, d. h. unendlich oft differenzierbar. In Abschnitt 3.2.2 wird die erste Ableitung der Ausgangsfunktion zur Berechnung einer *Lernregel* benötigt, daher sei sie hier der Vollständigkeit halber angegeben. Die Ableitung des Tangens Hyperbolicus ist durch

$$f'(x) = \frac{\mathrm{d}f}{\mathrm{d}x} = (1 - \tanh(x))^2 = (1 + f(x))(1 - f(x)) \tag{3.2}$$

definiert und ebenfalls in 3.1 abgebildet.

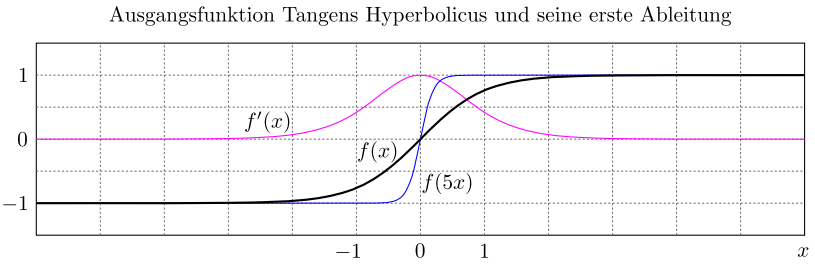

Abbildung 3.1: Die Ausgangsfunktion $f(x)$ (schwarz) mit steilem Anstieg $f(5x)$ (blau) und der ersten Ableitung $f'(x)$ (rot).

Senkt man die Gewichte soweit ab, dass sich das Eingangssignal mit einer Amplitude um ungefähr ±0,1 bewegt, so befindet sich das Signal in einem nahezu linearen Arbeitsbereich der Ausgangsfunktion. Linearisiert man den Tangens Hyperbolicus um die Nullstelle, d. h. bricht man die Taylorreihe nach dem zweiten Term ab, so ergibt sich die Näherung $\tanh(x) \approx \tanh(0) + \left(1 - \tanh^2(0)\right) \cdot x = x$ für $|x| \ll 1$. Diese Eigenschaft kann man sich beim Entwurf von Filterstrukturen mit neuronalen Netzen zunutze machen (vgl. dazu Abschnitt 5.1.4).

Bias

Für viele Anwendungen benötigt man einen voreingestellten Schwellwert, den sogenannten *Bias*. Dieser wird verwendet, um den Arbeitspunkt eines Neurons zu verändern oder um den Mittelwert der Eingangssignale auszugleichen. Der Bias kann entweder als zusätzlicher Term b_j zum effektiven Eingang dazu addiert werden oder einfach als ein weiterer Eingang o. B. d. A. $x_0 \equiv 1$ an das Neuron angelegt werden. Im letzteren Fall wird somit der Bias über das Gewicht w_0 eingestellt. In der grafischen Notation wird der Bias gelegentlich als Zahlenwert direkt in das Neuron geschrieben.

Eine kleine Amplitude vorausgesetzt, bewirkt eine Schwellwertanhebung (d. h. ein positiver Bias) eine Verschiebung des Arbeitspunktes in den logarithmischen Bereich der Ausgangsfunktion, wohingegen eine Absenkung den exponentiellen Bereich nutzbar macht. Vergleiche dazu die Kurvenform des Tangens Hyperbolicus in Abbildung 3.1.

Einzelneuron

Zusammenfassend ergibt sich nun das Gesamtmodell

$$y_j = \tanh\left(\sum_{i=1}^{n} w_{ji} x_i + b_j\right) \tag{3.3}$$

für ein einzelnes Neuron j. Dabei ist x_i das i-te Eingangssignal, welches über w_{ji} gewichtet und dann aufsummiert wird. Danach wird der Bias addiert, bevor schlussendlich die Ausgangsfunktion angewendet wird und den Ausgang y_j erzeugt.

Bei vielen Anwendungen spielt die Dimension der Zeit eine wichtige Rolle. Daher gibt es auch für sie eine Repräsentation innerhalb der neuronalen Architektur. In dieser Arbeit kommen *ausschließlich zeitdiskrete Modelle* zum Einsatz. Zum Vergleich ist im Anhang A.1.1 ein zeitkontinuierliches Neuronenmodell und der Zusammenhang zum zeitdiskreten Modell beschrieben. Betrachtet wird ein Neuron zum diskreten Zeitpunkt $t \in \mathbb{N}$, so ist die Aktualisierungsvorschrift durch

$$y_j(t+1) = \tanh\left(\sum_{i=1}^{n} w_{ji} x_i(t) + b_j\right) \tag{3.4}$$

gegeben. Die Grafik 3.2 fasst alle Bestandteile zusammen. Dabei kennzeichnet der Operator z^{-1} die Verzögerung des anliegenden Wertes um einen Zeitschritt.

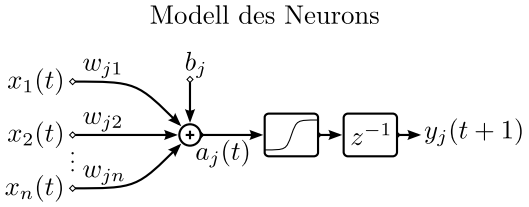

Abbildung 3.2: Das vollständige Modell des Einzelneurons: Die n verschiedenen mit w_{ji} gewichteten Eingänge x_i und der Bias b_j werden aufsummiert, durch die Ausgangfunktion f begrenzt und, je nach Anwendung, um einen diskreten Zeitschritt verzögert.

Erweitertes Synapsenmodell

Bisher wurden Synapsen als zeitlose Multiplikationen verstanden. Die Zeit wurde innerhalb der Neuronen als Verzögerung der Ausgabe bis zum nächsten Takt implementiert. Auch für Synapsen können zeitliche Abhängigkeiten formuliert werden. Dazu leitet man nicht nur das aktuelle Eingangssignal in das Neuron, sondern ergänzt es um zusätzliche zeitverzögerte Kopien desselbigen. Wahlweise kann man dann diese neuen Eingangssignale als weitere Eingänge betrachten, oder man fasst alle zu einer Quelle gehörenden Eingänge zu einer neuen Art Synapse zusammen. Abhängig vom Aufbau identifiziert man bei den Synapsen nun Filtereigenschaften, welche man je nach Charakteristik in verschiedene Klassen unterteilt. Abschnitt 5.1 stellt verschiedene Typen von Synapsen vor und erläutert deren Funktionsweise.

3.1.2 Übersicht über verschiedene Netzarchitekturen

Üblicherweise unterscheidet man in größeren Netzen zwischen *Ausgabeneuronen* und *verdeckten Neuronen*, je nachdem ob der Ausgang des Neurons nach außen gereicht wird oder lediglich netzintern weitergegeben wird. Oft organisiert man diese Neuronen auch in sogenannte *Schichten*. Speziell bei rein vorwärtsverknüpften (engl. *feed forward*) Netzen, spricht man von verdeckten Schichten und einer Ausgabeschicht. Die Aktivierungen durchlaufen in diesem Fall das Netz ausschließlich vorwärts, d. h. mit dem Signalfluss in Richtung der Ausgabeschicht.

Verbindet man Neuronen einer Schicht miteinander so spricht man von lateralen Verbindungen. Oft verwendet man dabei laterale Inhibition, um die Aktivierung der Neuronen in direkter Nachbarschaftsbeziehung gegeneinander abzugrenzen, z. B. zur Kantendetektion auf einer künstlichen Retina. Im Gegensatz zu rein vorwärtsverknüpften Netzen spricht man von *rekurrenten Netzen*, wenn es Neuronen mit lateralen oder rückwärtigen synaptischen Verbindungen, den sogenannten *Rekurrenzen*, gibt. Auch die *Selbstkopplung* eines Neurons zählt somit zu den Rekurrenzen. Die allgemeinste Form rekurrenter Netze sind vollständig verknüpfte Netze. Eine besondere Form rekurrenter Netze sind neuronale *Felder* [47]. Hierzu wird jedem Neuron noch ein Ort zugewiesen, anhand derer die Vernetzungsstruktur definiert wird. Dabei werden oft lokal erregende und global inhibierende Synapsen verwendet.

Einen ganz anderen Pfad schlägt *Reservoir Computing* [42, 43, 22] ein. Dabei erzeugt man ein vergleichsweise dünn verknüpftes rekurrentes Netz mit zufälligen, statischen Gewichten. Die Netzeingaben speist man über wenige Eingabeneuronen ein. Innerhalb des rekurrenten Netzes breitet sich nun eine komplexe nichtlineare Dynamik aus. Bei hinreichend vielen Neuronen erhält man somit ein reichhaltiges Reservoir, das man von außen durch gewichtete Verbindungen anzapfen kann. Beispielsweise produziert man dann aus einer Linearkombination der Anzapfungen ein Ausgangssignal.

3.2 Neuronale Lernregeln

Im folgenden Abschnitt werden neuronale Lernverfahren beschrieben, mit welchen die bisher als statisch angenommenen Verbindungsgewichte *gelernt* werden können. Dazu wird zu Beginn eine homöostatische Lernregel für ein einzelnes Neuron betrachtet. Danach wird ein allgemeines Trainingsverfahren für mehrschichtige Netze vorgestellt und der Abschnitt schließt mit einem selbstorganisierenden Verfahren.

Hinweis zur Notation: Der Übersichtlichkeit halber wird der Zeitindex weggelassen, sofern er nicht innerhalb einer Gleichung variiert oder für das Verständnis wichtig ist.

3.2.1 Homöostatische Plastizität

Unter homöostatischer (oder auch intrinsischer) Plastizität versteht man die Eigenschaft einzelner Neuronen ihre synaptischen Verbindungen in einer Selbstregulation derart anzupassen, dass eine Zielgröße in einem festgelegten Bereich bleibt [52, 49]. Man kann die homöostatische Plastizität damit zur Klasse der unüberwachten Lernverfahren zählen, weil in der Regel alles für die Regulation benötigte Wissen lokal am Neuron vorhanden ist bzw. dieses von ihm selbst erzeugt wird.

Konkret bedeutet Lernen hier, dass ein Neuron die Gewichte seiner Eingangssynapsen und den Bias selbständig den Eingangssignalen anpasst. Die Zielgröße ist dabei *maximaler Informationstransfer* durch das Neuron. Sind die Eingangssignale zu groß, führt das vermehrt zu einer Signalsättigung bedingt durch die Ausgangsfunktion. Sind die Signale dagegen zu klein, können sie sich nur schlecht vom Rauschen konkurrierender Eingangssignale abheben. Besitzt das Signal einen hohen Mittelwert, so ist der volle Umfang des Signals blockiert, indem eine Halbwelle stetig in die Sättigung der Ausgangsfunktion geschoben wird. Der homöostatische Prozess wird demnach so gestaltet, dass das Ausgangssignal möglichst ausgeglichen und aussagekräftig ist. Betrachtet man die Ausgangsfunktion des Neurons als eine Art Fenster, so wird das Signal mit seiner eigentlichen Dynamik genau so skaliert, dass möglichst viel der Information hindurch gelangen kann. Dabei ist es wichtig, die Sättigung zu vermeiden und die Sensitivität zu erhöhen, um *interessante* Bereiche des Eingangssignals hervorzuheben.

Die homöostatische Plastizität führt nachweislich zu einer verbesserten Signalpropagierung [52]. Denkbar ist auch der Einsatz solch einer Lernregel für Eingangsneuronen, deren sensorische Eingaben eine nichtstationäre Dynamik haben und in Abhängigkeit des sensomotorischen Kontexts eine andere Gewichtung benötigen. Erhöhte Sensitivität verstärkt auch vorhandenes Rauschen, was in diesem Falle aber weniger als Problem zu verstehen ist. Vielmehr bringt es die Möglichkeit vorhandene Symmetrien zu brechen und mögliche lokale Minima im Lernvorgang wieder zu verlassen. Ein wohldosierter Grad an Zufälligkeit ist für viele Lernregeln förderlich – wenn nicht sogar notwendig.

Infomax-Lernregel

Eine mögliche Realisierung einer Lernregel, welche die oben genannten Eigenschaften besitzt, ist die Infomax-Lernregel [2, 3]. Betrachtet wird hier die Anwendung auf ein einzelnes Eingabeneuron

$$y = \tanh(wx + b)$$

mit Eingangsgewicht w und Bias b. Gesucht werden also die Gewichtsänderungen im Hinblick darauf, die Zielgröße Informationstransfer (genauer: die *Transinformation* I) zu maximieren. Zur Herleitung der Gewichtsänderung aus der Zielgröße wird ein Gradientenverfahren verwendet. Die vollständige Herleitung der Infomax-Lernregel, für das in dieser Arbeit verwendete Neuronenmodell, ist aufgrund ihrer Länge im Anhang A.1.2 zu finden. Dort ist beschrieben, was formell unter dem Begriff Informationstransfer zu verstehen ist und wie daraus eine Lernregel zu dessen Maximierung abgeleitet werden kann. Dieser Abschnitt reduziert sich daher auf die Erläuterung der Funktionsweise der Lernregel.

Die resultierenden Lernregeln für die Gewichte w und b sind

$$\Delta w = \eta_w \left(\frac{1}{w} - 2xy \right) \tag{3.5}$$

$$\Delta b = -\eta_b \, y \tag{3.6}$$

mit den Lernraten $0 < \eta_{w,b} \ll 1$. Die Gewichtsänderungen Δw und Δb werden dann

über den üblichen Korrekturschritt

$$w(t+1) = w(t) + \Delta w(t)$$
$$b(t+1) = b(t) + \Delta b(t)$$

angewendet. Die Gleichung (3.6) ist nur vom Ausgang des Neurons abhängig und entfernt den konstanten Anteil des Signals x, indem es den Bias auf den negativen Mittelwert von y einstellt. Gleichung (3.5) kann gedanklich in zwei Teile zerlegt werden. Der erste Teil versucht unentwegt den Eingang zu verstärken, indem er das Gewicht erhöht. Dies geschieht umso langsamer, je größer das Gewicht schon ist. Die zweite Hälfte ist die Gegenkraft, welche das Gewicht absenkt, wenn sowohl das Eingangssignal x als auch das Ausgangssignal y zu groß werden. In Abbildung 3.3 ist die Wirkung der angegebenen Lernregeln auf den Ausgang eines Einzelneurons gezeigt. Die ersten beiden Graphen zeigen das nichtstationäre Eingangssignal x und das Ausgangssignal u ohne eine Anpassung. Darunter ist das Ausgangssignal y mit intrinsischer Plastizität der Gewichte abgebildet. Die Startwerte für die Gewichte sind hierbei $w_0 = 1$ und $b_0 = 0$.

Experiment

Das (Test-)Eingangssignal ist eine mit leichtem Rauschen überlagerte Sinusschwingung mit konstantem Signalanteil. Nach etwa 40 Sekunden verändert sich das Eingangssignal indem sich der Mittelwert absenkt. Um einen Sensorausfall zu imitieren, fällt nach 120 Sekunden das Eingangssignal sogar aus. Übrig bleibt nur das Rauschen. Nach weiteren 40 Sekunden ist das Signal wieder da, wobei es aber wieder um den vorherigen Mittelwert schwingt.

Wie man erkennt, verharrt das unangepasste Ausgangssignal weit in der Sättigung und wird daher nur unzureichend übertragen, hier wäre mindestens eine manuelle Kalibrierung erforderlich. Allerdings muss dazu in jedem Fall klar sein, welchen Bereich das Eingangssignal im Äußersten abdeckt. Im dritten und vierten Graphen ist die Reaktion der Lernregel auf das Eingangssignal zu sehen. Die Lernrate beträgt $\eta = 0{,}001$. Anfangs wird das Eingangsgewicht w herunterskaliert und der Bias abgesenkt. Ab Sekunde 40 wird die Änderung des Eingangs detektiert und der Bias wird wieder angehoben. Bei Sekunde 120 verstärkt die Lernregel das Eingangssignal erheblich. Somit bestünde die Chance noch Informationen in dem verbleibenden Signal, in diesem Fall Rauschen, zu finden. Sobald das ursprüngliche Signal wieder eingeschaltet ist, pegelt sich das Gewicht nach weniger als 10 Zeitschritten wieder in den normalen Bereich ein. Durch das zeitweilig erhöhte Eingangsgewicht ist allerdings auch Information verloren gegangen.

In Abbildung 3.4 sind die relativen Häufigkeiten der Signale x, u und y für die ersten 40 Sekunden des Tests im Vergleich. Für das Eingangssignal erkennt man die für eine Sinus-Schwingung typische Badewannenform der Verteilung. Das Histogramm des unangepassten Ausgangssignals ist erwartungsgemäß entartet. Die meisten Werte sind in der positiven Sättigung bei +1. Für das angepasste Ausgangssignal ist die Ähnlichkeit zur Eingangsverteilung deutlich erkennbar.

Doch es ergibt sich ein weiterer interessanter Effekt. Wie aus der Herleitung (siehe Anhang A.1.2) ersichtlich, wird durch Anwendung der Lernregeln die Transinformation

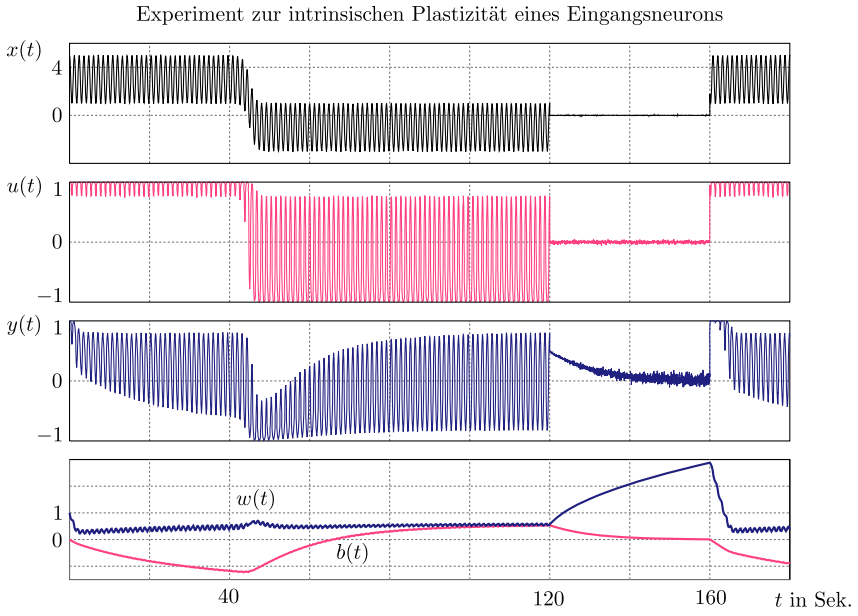

Abbildung 3.3: Experiment zur intrinsischen Plastizität eines Einzelneurons. Das Signal $x(t)$ stellt ein nicht-stationäres Eingangssignal mit Unterbrechung dar (schwarz). Daraufhin würde ein Neuron mit statischen und schlecht voreingestellten Gewichten ein unbrauchbares Ausgangssignal wie $u(t)$ erzeugen (rot). Der interessante Aspekt des Eingangssignals sei die Schwingung, welche durch die sichtbaren Störungen nur unzureichend propagiert werden kann. Das adaptive Neuron produziert hingegen das Signal $y(t)$ und passt sich den gegebenen Eigenheiten des Signals an (blau). Die letzte Grafik zeigt den Verlauf der Gewichte.

durch das Neuron maximiert. Das ist äquivalent zur Maximierung der *Ausgangsentropie* des Neurons. Die Entropie ist bei gleichverteilten Signalen maximal, d. h. die Lernregel ist versucht, die Entropie des Ausgangs zu maximieren und überführt, so weit es geht, die Verteilung der Eingangswerte in eine Gleichverteilung. Dies ist im Ansatz an den Rändern der Verteilung von y zu beobachten.

Für stark geglättete und schwach variierende Signale ist die Infomax-Lernregel leider gänzlich ungeeignet. Nachdem der Mittelwert ausgeglichen ist, wird durch unaufhaltsames Verstärken des Eingangsgewichts vergeblich versucht eine nicht vorhandene Dynamik zu erreichen. Daher sollte eine angemessene obere Schranke für das Eingangsgewicht festgelegt werden, welche bei Überschreiten zugleich auch als Detektor für evtl. sensorische Ausfälle fungiert.

Fazit

Für die Verarbeitung nichtstationärer (Sensor-)Signale kann zur Laufzeit die Infomax-Lernregel angewendet werden, um die Dynamik des Signals auf den *nützlichen* Bereich einzustellen. Hierfür gleicht sie zugunsten einer höheren Dynamik den Mittelwert des Signals aus, da dieser im Allgemeinen wenig Information trägt. Dabei vermeidet sie störende Signalsättigungen und maximiert den statistischen Informationsgehalt des

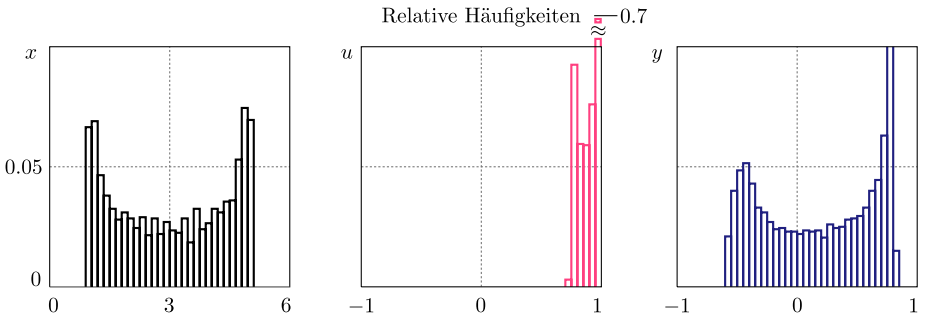

Abbildung 3.4: Die Abbildung zeigt die Histogramme für die ersten 40 Sekunden des Eingangssignals (schwarz), des unadaptierten (rot) und des adaptiven Ausgangs (blau). Die Lernregel stellt die ursprüngliche Verteilung wieder her und versucht sie darüber hinaus einer Gleichverteilung anzunähern.

Ausgangssignals. Bei sensorischer Deprivation erhöht die Infomax-Lernregel die Sensitivität und verstärkt in dem vorhandenen Signal noch verwertbare Informationen. Die Lernregel arbeitet selbstregulativ und stabil, vorausgesetzt die Lernrate ist hinreichend klein, damit die Zeitskala der Regelung weit langsamer als die Dynamik des Signals ist.

Die Lernregel ist zudem sparsam mit konstantem Rechenzeit- und Speicherverbauch. Sie arbeitet nur auf den lokalen und aktuellen Werten von Ein- und Ausgang und braucht keinen zusätzlichen Speicher. Die teuerste arithmetische Operation ist die Division. Für zeitkritische Anwendungen wäre zu überlegen, die Kehrwertfunktion $f(w) = w^{-1}$ zu approximieren und sie dabei beispielsweise ab $w > 8$ zu Null zu runden, um automatisch ein weiteres Anwachsen des Eingangsgewichts bei Signalausfall zu verhindern. Das ist sinnvoll, da auch Regulationsprozesse gewissen Sättigungen unterliegen, was in diesem Fall eine Art Schutzfunktion vor grenzenloser Überregulierung ist.

3.2.2 Fehlerrückführung

Ein bewährtes Lernverfahren für vorwärtsverknüpfte Netze ist *backpropagation of error*, zu deutsch: Fehlerrückführung [37]. Das Verfahren gehört zur Klasse der überwachten Lernverfahren, d. h. es gibt einen Lehrer, welcher dem Netz eine Aufgabe stellt und zugleich die richtige Lösung präsentiert. Die vom Netz errechnete Lösung wird mit der des Lehrers verglichen und die Abweichung als *Fehler* an das Netz zurückgegeben. Mit der Information über diesen Fehler können die synaptischen Verbindungen in der Art neu eingestellt werden, dass bei der nächsten Berechnung die Abweichung von Netzausgang zum Lehrersignal geringer ausfällt. Die Aufgabe, die ein neuronales Netz dabei zu lösen hat, ist zu einer gegebenen Netzeingabe eine vom Lehrer erwartete Netzausgabe zu errechnen. Ein Paar, bestehend aus den Netzeingaben und den dazugehörigen erwarteten Netzausgaben, heißt Trainingsbeispiel.

Klassischerweise werden neuronale Netze mit *backpropagation* in Episoden trainiert (sogenanntes *batch learning*), d. h. es werden nacheinander die Trainingsbeispiele abgearbeitet und die dabei festgestellten Abweichungen aufsummiert. Erst am Ende einer

Episode werden die Gewichte neu eingestellt und eine neue Episode gestartet. Dieser Vorgang wird normalerweise wiederholt, bis die Abweichungen wunschgemäß klein sind oder der Prozess stagniert und der Fehler nicht mehr weiter zu senken ist. Diese Art des Trainings eignet sich gut, wenn eine ausreichend repräsentative Menge an Trainingsbeispielen zur Verfügung steht, welche idealerweise alle Facetten der Aufgabenstellung gleichmäßig abbilden. Gibt es aber beispielsweise zu wenig oder zu einseitig ausgerichtete Trainingsbeispiele lernt das Netz bei zu langem Training die Beispiele mehr oder weniger auswendig und kann schlecht *generalisieren*. Um zu testen ob ein Netz auch wirklich gut generalisiert, kann man eine ausreichende Menge an Trainingsbeispielen in einen Trainingssatz und einen Testsatz aufteilen. Nachdem auf dem Trainingssatz einige Episoden trainiert wurden und der Fehler klein genug ist, wird mit dem Testsatz überprüft, ob ein vergleichbares Ergebnis erzielt werden kann. Mit *batch learning* können größere Netze vorab trainiert werden, um dann für eine konkrete Anwendung, wie zum Beispiel für die Handschrifterkennung, eingesetzt zu werden.

Stehen zu Beginn des Trainings keine Trainingsbeispiele zur Verfügung, d. h. können diese erst zur Laufzeit des Netzes erzeugt werden, so muss *online* gelernt werden. Das heißt konkret, dass nach jeder Netzausgabe unmittelbar eine Anpassung der synaptischen Verbindungen vorgenommen wird. Das *online backpropagation* eignet sich somit gut für eine unbestimmte oder unbegrenzte Trainingsdauer.

Backpropagation-Algorithmus

Der Algorithmus besteht aus den folgenden drei Phasen.

1. Anlegen der Netzeingaben und durchrechnen der Aktivierungen bis zur Ausgabe.
2. Vergleichen der Netzausgabe mit dem Lehrersignal und Berechnen des Fehlers.
3. Rückführen des Fehlers zu den einzelnen Neuronen und Anpassen der Gewichte.

Der Gesamtfehler, d. h. der über alle Ausgangsneuronen aufsummierte quadratische Fehler des Netzes ist definiert als

$$E = \frac{1}{2} \sum_j (d_j - y_j)^2 \,, \tag{3.7}$$

wobei $\mathbf{y}$ die Netzausgaben und $\mathbf{d}$ die korrespondierenden Lehrererwartungen sind. Damit der Fehler des Netzes abnimmt, müssen aber *genau* die Gewichte justiert werden, die einen Beitrag zum Fehler geliefert haben. Jedes Gewicht wird daher proportional zu *seinem* Anteil am Gesamtfehler korrigiert. Um diesen Anteil zu errechnen wird der Fehler E partiell zu jedem Gewicht w_{ji} abgeleitet. Die Gewichtsänderung

$$\Delta w_{ji} = -\eta \frac{\partial E}{\partial w_{ji}} \tag{3.8}$$

wird dann im Korrekturschritt

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}(t) \tag{3.9}$$

angewendet. Die gesamte Lernregel leitet sich wie folgt her. Zuerst zerlegt man die partielle Ableitung aus 3.8 mit Hilfe der Kettenregel der Differentialrechnung in

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial a_j} \frac{\partial a_j}{\partial w_{ji}}$$

und rechnet die einzelnen Faktoren aus. Der erste Faktor ist $\partial E / \partial y_j = -(d_j - y_j)$. Glücklicherweise kürzt[2] sich die 2 von der Ableitung des quadratischen Terms heraus, weil in Gleichung (3.7) ein $1/2$ vorangestellt ist. Der zweite Faktor ist nichts weiter als die Ableitung der Ausgangsfunktion $y'_j = f'(a_j)$, in unserem Fall der Tangens Hyperbolicus, dessen Ableitung wir bereits in Gleichung (3.2) ausgerechnet haben. Die verbleibende partielle Ableitung $\partial a_j / \partial w_{ji}$ mit $a_j = \sum_i w_{ji} x_i$ hat als Ergebnis schlicht die Eingaben x_i von Neuron j. Somit ergibt sich die *backpropagation*-Lernregel bezogen auf ein einzelnes Gewicht w_{ji} durch

$$\Delta w_{ji} = \eta \, (d_j - y_j)(1 + y_j)(1 - y_j) \, x_i \tag{3.10}$$

vorerst für einschichtige Netze.

Der Faktor $\eta \in \mathbb{R}$ ist die Lernrate. Sie bestimmt die Geschwindigkeit der Adaption und ist eine kleine positive Konstante $0 < \eta \ll 1$. Nicht selten verwendet man eine im Verlauf des Trainings absinkende Lernrate z. B.

$$\eta(t) = e^{-\beta t} \tag{3.11}$$

mit $\beta > 0$, welche die Lösung in den Minima der Fehlerfunktion (3.7) besser stabilisiert. Solche Verfahren sind unter *simulated annealing* bekannt.

Die Lernrate ist stets ein kritischer Parameter. Wird sie zu gering eingestellt, so ist das Lernen sehr träge und braucht unnötig viele Lernzyklen, jedoch ist das Lernen auch robust gegen Rauschen auf dem Trainingssignal. Bei zu groß gewählter Lernrate kann bei einem stark verrauschten Trainingssignal der ganze Lernalgorithmus instabil werden. Beobachtet man entdämpfte Oszillationen auf dem zeitlichen Verlauf des mittleren quadratischen Fehlers oder dem Verlauf der Gewichte, so muss die Lernrate verkleinert werden. Die *richtige* Wahl der Lernrate ist stark von der Aufgabenstellung und den Trainingsdaten abhängig. Somit kommt nicht selten der Wunsch nach einer adaptiven Lernrate auf.

Erweiterung auf mehrschichtige Netze

Für die Anwendung auf mehrschichtige Netze muss die Lernregel erweitert werden, da für Neuronen aus verdeckten Schichten kein direktes Lehrersignal existiert. Daher muss eines errechnet werden. Dazu nimmt man ein Neuron j aus einer verdeckten Schicht und ermittelt den indirekten Fehler E_j, den dieses Neuron zum Gesamtversagen des ganzen Netzes beigetragen hat. Dieser Fehler ist

$$E_j = \sum_k \delta_k w_{kj} \tag{3.12}$$

[2] Bei der Aufstellung von Lernregeln ist es üblich, dass alle konstanten Faktoren in die Lernrate eingehen und somit zu einer Konstanten zusammengefasst werden.

also einfach die Fehlersumme der von diesem Neuron aktivierten Folgeneuronen – jeweils proportional zu dem synaptischen Gewicht, welches jeweils zwei Neuronen miteinander verbindet. Kürzt man die Lernregel auf

$$\Delta w_{ji} = \eta \delta_j x_i \tag{3.13}$$

ab, so muss man für jedes Neuron entscheiden ob:

$$\delta_j = \begin{cases} (1+y_j)(1-y_j)(d_j - y_j) & \text{wenn } j \text{ ein Ausgabeneuron ist,} \\ (1+y_j)(1-y_j)\sum_k \delta_k w_{kj} & \text{wenn } j \text{ verdecktes Neuron ist.} \end{cases} \tag{3.14}$$

Zum *backpropagation*-Verfahren wurden viele Erweiterungen entwickelt, meistens für die Anwendung im *batch learning*. [39] und [29] geben eine Übersicht und vergleichen die Leistungsfähigkeit der Erweiterungen. In der folgenden Arbeit kommt aber, wenn nicht explizit anders angegeben, das *online backpropagtion* im Grundzustand mit vorerst konstanter Lernrate zum Einsatz, da es vergleichsweise robust ist, wenig Rechenzeit- und Speicherressourcen benötigt und mit nur einem Parameter auskommt. In Abschnitt 5.1.4 wird dazu exemplarisch ein Netz trainiert.

Die Wahl der Zielfunktion

In Gleichung (3.7) wurde als zu optimierende Größe die Minimierung der quadratischen Abweichung von Lehrersignal und Netzausgang angenommen. Das Verfahren kann natürlich auch auf andere *Zielfunktionen* angewendet werden. In [21] wurde beispielsweise ein Netz als Umschaltgatter für andere Netze trainiert. Dieses Netz soll nun anhand der Eingaben entscheiden, welches der anderen Netze wohl für die gegebenen Eingaben die besten Wahl ist und diese dann an den Ausgang *weiterleiten*. Die vermeintlich schlechteren Netze werden vom Ausgang abgetrennt. Das Umschaltgatter wird also darauf spezialisiert, nur *einen* Netzausgang weiterzuleiten – und möglichst den für die Aufgabe am besten geeignetsten.

Gradientenverfahren für rekurrente Netze

Rekurrente neuronale Netze sind ein mächtiges aber mitunter schwer handhabbares Werkzeug. Sie sind in der Lage komplizierte Dynamiken bis hin zu chaotischen Zeitreihen zu erzeugen. Es konnte gezeigt werden, dass vorwärtsverknüpfte Netze mit nur einer verdeckten Schicht im Prinzip beliebige Funktionen approximieren können und dass vollständig verknüpfte rekurrente Netze als Modelle für beliebige dynamische Systeme verwendet werden können [6]. Daher liegt es nahe die bestehenden Lernverfahren auch für rekurrente Netze zu erweitern. Die bekanntesten Vertreter klassischer Gradientenverfahren für rekurrente Netze sind *Real Time Recurrent Learning* (RTRL) [53] und *Backpropagation Through Time* [50]. Eine gute Übersicht über Lernverfahren für rekurrente Netze findet man in [34, 8].

Das Training rekurrenter neuronaler Netze stellt sich aber im Verhältnis zu reinen vorwärtsverknüpften Netzen als wesentlich schwieriger heraus. Um den exakten Gradienten zu berechnen sind die meisten bisherigen Verfahren sehr rechenaufwendig. So benötigt beispielsweise das RTRL eine Rechenzeit der Komplexität $\mathcal{O}(n^4)$ und hat

einen mit $\mathcal{O}(n^3)$ skalierenden Speicherverbrauch, wobei n hier für die Anzahl der Neuronen steht. Für Anwendungen in denen die Rechenressourcen knapp bemessen sind scheidet solch ein Verfahren frühzeitig aus.

Der ausschlaggebende Punkt, dass Gradientenverfahren für rekurrente Netze Schwierigkeiten bereiten können, ist jedoch ein ganz anderer. Während des Lernens durchläuft die Netzdynamik durch die Veränderung der Gewichte mitunter einen kritischen Punkt im Zustandsraum (eine sogenannte *Bifurkation*), wobei sich das Verhalten des Netzes drastisch ändern kann. Für die genannten Gradientenverfahren kann das im schlimmsten Fall bedeuten, dass diese *instabil* werden, weil durch abrupte Änderungen im Verlauf die Gradienten plötzlich besonders groß werden. Es wurden Ansätze aufgezeigt, wie solche Schwierigkeiten umgangen werden können [48]. Verzichtet man auf den exakten Gradienten wird man durch ein stabileres Lernverfahren belohnt. Dazu behandelt man die rekurrenten Verbindungen als wären es übliche Netzeingänge [8] und lernt klassisch mit *backpropagation*. Ein anderer Ansatz ist eine restriktivere Netztopologie. Man verzichtet auf vollständige Verknüpfung und lässt nur lokale Rekurrenzen zu, für welche dann spezielle Lernregeln hergeleitet werden können und deren Stabilität beweisbar ist [36]. Für einige Architekturen [10] verwendet man auch statische (d. h. nicht lernende) rekurrente Verbindungen [42, 22] oder erzeugt Rekurrenzen in Form sogenannter *Kontextneuronen* (vgl. dazu Abschnitt 5.1.5).

3.2.3 Wachsendes Neuronales Gas

Weitere Formen des unüberwachten Lernens (engl. *unsupervised learning*) sind *Selbstorganisierende Netze* und *Neuronale Gase*. Deren gemeinsame Grundidee ist der unüberwachte, sukzessive Aufbau einer topologischen Struktur oder *Karte*, welche den sensorischen bzw. sensomotorischen Eingaberaum repräsentiert.

Dabei wird der Eingaberaum als Wahrscheinlichkeitsdichte verstanden und versucht, eine oft begrenzte Menge an Neuronen (auch Knoten oder Einheiten) so im mehrdimensionalen Eingaberaum zu platzieren, dass dieser der Wahrscheinlichkeitsdichte entsprechend *optimal* abgedeckt ist. Dabei stellt jedes Neuron eine Art Repräsentant für eine Menge von Eingabedaten dar. Die möglicherweise hochdimensionalen und (quasi-) kontinuierlichen Eingabedaten werden somit auf eine vergleichsweise geringe Anzahl von Repräsentanten abgebildet. Der Eingaberaum wird damit in gewisser Weise diskretisiert. Der Erfolg eines bestimmten Verfahrens ist stark abhängig von der Wahl der Parameter und der Stationarität der Eingabedaten. Weiterhin wird während des Lernens die topologische Struktur der Eingabedaten herausgearbeitet, indem zwischen benachbarten Neuronen Synapsen (d. h. Kanten) aufgebaut werden.

Der Algorithmus *Growing Neural Gas with Utility Criterion* (GNG-U) [12, 13, 14] baut eine solche topologische Struktur durch eine Form des Hebbschen Lernens auf. Es wird zu jedem Eingabevektor das Neuron bestimmt, welches dem Vektor am nächsten liegt. Dabei werden die quadratischen Abweichungen des Eingabevektors zu den Gewichtsvektoren aller Neuronen miteinander verglichen. Das Neuron mit dem geringsten Abstand gewinnt und wird samt seiner über bestehende Kanten definierten Nachbarschaft anderer Neuronen in Richtung des Eingabevektors adaptiert. Zwischen den besten beiden Neuronen wird eine neue Kante erzeugt bzw. das fortschreitende Alter wieder zurückgesetzt, womit die Kante als *kürzlich in Gebrauch* markiert ist.

Dieser spezielle Typ von Synapse ist von Beginn an ungerichtet und ungewichtet. Ihre einzige Eigenschaft ist ihr Alter, wobei häufig verwendete Synapsen verjüngt werden und gealterte Synapsen irgendwann *absterben*.

Das Verfahren beginnt mit nur zwei Neuronen und fügt sukzessive neue Neuronen ein. Diese werden an den Stellen mit hoher Fehlerdichte eingefügt, um den Gesamtfehler des Systems zu minimieren. Speziell bei nicht-stationären Daten ist es somit irgendwann erforderlich die nicht mehr verwendeten Neuronen wieder zu entfernen. Das ist nötig, um im allgemeinen Fall begrenzter Rechenzeit und Speicherkapazität Ressourcen freizugeben, damit das Verfahren wieder Neuronen an *wichtigeren* Stellen einfügen kann.

Der vollständige formale Algorithmus lautet wie folgt und die standardmäßig verwendeten Parameter sind in Tabelle 3.1 gegeben.

Der GNG-U–Algorithmus

0. Beginne mit nur zwei Einheiten a und b. Wähle die Gewichte $\mathbf{w}_a$ und $\mathbf{w}_b$ zufällig aus $\mathbb{R}^D$, wobei $D \in \mathbb{N}$ die Dimension der Eingabedaten ist.

1. Erzeuge einen Eingabevektor ξ bzw. wähle zufällig einen aus einer Trainingsmenge.

2. Bestimme die Einheit s_1, welche dem Eingabevektor am nächsten ist bzw. die zweitnächste Einheit s_2.

3. Erhöhe das Alter aller von s_1 ausgehenden Kanten.

4. Addiere die quadratische Abweichung der Einheit s_1 zum Eingabevektor auf eine lokale Fehlervariable E_{s_1} und berechne die Nützlichkeit U_{s_1} der Einheit s_1.

$$\Delta E_{s_1} = \|\mathbf{w}_{s_1} - \xi\|^2 \tag{3.15}$$
$$\Delta U_{s_1} = \Delta E_{s_2} - \Delta E_{s_1} \tag{3.16}$$

5. Adaptiere die Gewichte der Einheit s_1 und ihrer direkten topologischen Nachbarn. Die Adaption folgt unmittelbar in Richtung des aktuellen Eingabevektors. Die Stärke der Adaption von s_1 regelt die Lernrate $\varepsilon_b \in \mathbb{R}$. Für alle $n \in \mathbb{N}$ Nachbarn von s_1 wird die geringere Lernrate $\varepsilon_n \in \mathbb{R}$ verwendet.

$$\Delta \mathbf{w}_{s_1} = \epsilon_b(\xi - \mathbf{w}_{s_1}) \tag{3.17}$$
$$\Delta \mathbf{w}_n = \epsilon_n(\xi - \mathbf{w}_n) \tag{3.18}$$

6. Wenn bereits eine Kante von s_1 zu s_2 besteht setze das Alter zurück bzw. erzeuge eine neue Kante, falls eine solche bisher noch nicht existiert.

7. Entferne alle Kanten deren Alter den Schwellwert $a_{max} \in \mathbb{N}$ überschritten haben. Falls das dazu führt, dass von einer Einheit aus keine Kanten mehr ausgehen, entferne diese ebenfalls.

8. Wenn die Anzahl der Zyklen ein Vielfaches der Zahl $\lambda \in \mathbb{N}$ durchschreitet dann füge wie folgt eine neue Einheit ein:

a) Bestimme die Einheit q, welche den maximalen akkumulierten Fehler aufweist.

$$q = \arg\max_c E_c \tag{3.19}$$

b) Füge eine neue Einheit r zwischen q und seinem Nachbarn f ein, der den größten akkumulierten Fehler aus der Nachbarschaft von q hat. Initialisiere das Gewicht von r wie folgt:

$$\mathbf{w}_r = 0{,}5\,(\mathbf{w}_q + \mathbf{w}_f) \tag{3.20}$$

c) Füge zwei neue Kanten ein, die r jeweils mit q und f verbinden und entferne die ursprüngliche Kante zwischen q und f.

d) Reduziere die Fehlervariablen E_q und E_f um den Faktor α und initialisiere die neue Fehlervariable und die Nützlichkeit von r wie folgt:

$$E_r = \frac{(E_q + E_f)}{2} \tag{3.21}$$

$$U_r = \frac{(U_q + U_f)}{2} \tag{3.22}$$

9. Reduziere alle akkumulierten Fehler und Nützlichkeiten durch Multiplikation mit einer Konstanten $d \in \mathbb{R}$, $0 < d < 1$.

10. Entferne unnütze Einheiten indem die Einheit i mit der geringsten Nützlichkeit

$$i = \arg\min_c U_c \tag{3.23}$$

bestimmt wird und falls die Bedingung

$$E_q / U_i > k \tag{3.24}$$

erfüllt ist, lösche alle von i ausgehenden Kanten und entferne i.

11. Falls kein Abbruchkriterium erfüllt ist, kehre zurück zu Schritt 1. Solch ein Abbruchkriterium könnte z. B. die maximale Netzgröße N_{max} sein oder bei stationären Eingabedaten das Unterschreiten eines bestimmten mittleren Fehlers aller Neuronen.

a_{max}	ϵ_b	ϵ_n	d	α	λ	k
50	0,2	0,006	0,995	0,5	100	3

Tabelle 3.1: Die Standardparameter des GNG-U nach [12, 13].

Implementation und Test

Die Implementierung des GNG-U ist dank der sehr detaillierten Beschreibung in vielen Hochsprachen einfach zu bewerkstelligen und mit dem gegebenen Parametersatz hat man für viele Anwendungen einen guten Ausgangspunkt für die eigenen Experimente gegeben. Gänzlich ohne eine Anpassung der Parameter kommt man allerdings nicht aus. Besonders bei nicht-stationären Daten sind durch eine behutsame Anpassung der Parameter verschiedene *Reaktionen* des Netzes auf die Veränderung der Daten präzise einzustellen. Driften Mittelwert oder Varianz der Eingabedaten nur langsam weg, so reicht mitunter eine Anpassung der Lernraten, sodass die Struktur des Netzes erhalten bleibt und sich nur die einzelnen Knoten verschieben. Ändert sich allerdings die grundlegende Struktur der Eingabedaten, so kann durch Anpassung des Kantenalters oder der Löschungsbedingung aus Schritt 11 eine schnelle Umstrukturierung des Netzes erzielt werden.

In Abbildung 3.5 sind interessante Zwischenschritte eines Experiments zur Verifikation abgetragen. In jedem Zeitschritt wird ein zufälliges Eingabedatum ξ eingespeist. Die letzten 100 Eingabedaten sind als blaue Punkte in die Grafik eingezeichnet, wobei die älteren Punkte langsam ausgeblendet werden. Zu Beginn des Experiments fallen die Eingabedaten in einen kleinen Kreis. Nach 2000 Zeitschritten wird die Struktur der Eingabedaten schlagartig auf eine Spirale umgeschaltet. Sofort ist die Reaktion des Netzes zu beobachten und nach weiteren 1000 Zeitschritten haben sich die meisten bestehenden Knoten bereits grob über die neu aufgespannte Fläche verteilt, können aber die feine Struktur der Spirale noch nicht adäquat wiedergeben. Im weiteren Verlauf werden nach und nach neue Knoten akquiriert und unnütze Kanten entfernt. Das Alter der Kanten entscheidet in Schritt 7 über ihre mögliche Ausrationalisierung. Ein hohes Kantenalter wird in der Grafik durch eine rote Kante dargestellt. Zum Zeitschritt $t = 6000$ ist das Netz schon fast vollständig adaptiert und die letzten unnützen Kanten sind bereits durch ein hohes Alter markiert. Nun wird wiederum abrupt die Struktur der Eingabedaten auf den alten Zustand zurückgesetzt. Einige Knoten im Innern der Spirale werden daraufhin nur leicht verschoben und können wiederverwendet werden. Der Großteil der Knoten wird allerdings durch das Nützlichkeitsmaß eingeholt und nach und nach abgebaut.

Fazit

Wachsende neuronale Gase bilden eine ausgezeichnete Ausgangsbasis für die unüberwachte Aufteilung unbekannter sensorischer Zustandsräume. Dabei passt sich das Netzwerk selbständig an die Verteilung der Eingabedaten an und erzeugt sogleich eine topologische Struktur der Daten, welche dabei hilfreich ist Zusammenhänge zwischen den durch die Knoten repräsentierten Kategorien zu identifizieren. Der kontinuierliche Eingabestrom wird in diskrete Kategorien aufgebrochen, wobei die Auflösung der Kategoriebildung von der Dichte der Eingabedaten abhängt.

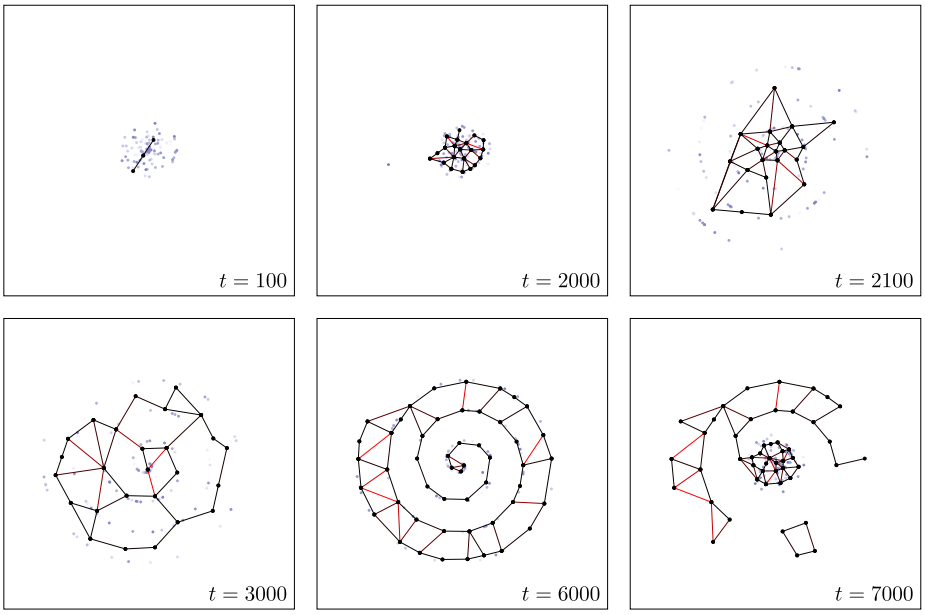

Abbildung 3.5: Abgebildet sind sechs Zwischenzustände eines typischen Verlaufs eines GNG-U-Experiments unter Verwendung der in Tabelle 3.1 angegebenen Parameter. Zu den Zeitpunkten $t_1 = 2000$ und $t_2 = 6000$ wechselt die Verteilung der zufälligen Eingabedaten, worauf sich das neuronale Gas nach einigen Zeitschritten durch eine Umstrukturierung anpasst.

4 Modell des Individuums

Nachdem nun die grundlegenden Konzepte und Werkzeuge beschrieben wurden, muss das Modell definiert werden. Innerhalb dieses Kapitels wird das Rahmenwerk des Individuums in Anlehnung an [32] beschrieben. Dabei werden die einzelnen Bestandteile als Module aufgefasst, deren Schnittstellen definiert sowie die Wechselwirkungen zwischen den Modulen diskutiert. Wie genau die einzelnen Module implementiert werden, erläutern die anschließenden Kapitel 5, 6 und 7. Dieses Kapitel ist wie folgt aufgebaut: Der erste Abschnitt 4.1 beschreibt die sensorischen und motorischen Schnittstellen zu Körper und Umwelt. Darauf folgen Abschnitte zur selbstorganisierten Aufteilung des Zustandsraums (4.2) und deren Verwaltung auf unbestimmte Dauer (4.3). Weiter geht es mit der Beschreibung explorativen Verhaltens und der Definition von Lernfortschritt in Abschnitt 4.4. Der letzte Abschnitt 4.5 beschreibt die Bewertung und Auswahl motorischer Aktionen und das Kapitel schließt mit einer Zusammenfassung des gesamten Modells anhand einer Übersichtsgrafik.

4.1 Der sensomotorische Apparat

Die Grundvoraussetzung für ein Modell vom Lernen auf unbestimmte Dauer, ist ein geschlossener Zyklus aus Wahrnehmen, Denken und Handeln. Wahrzunehmen sind hierbei unterschiedliche Eigenschaften des Körpers bzw. der Umwelt. Dazu wandeln Sensoren diese Eigenschaften in für das System verarbeitbare numerische Werte um. Für das lernende Individuum ist die Beschaffenheit, Qualität oder Herkunft dieser Umweltsignale vorerst nicht von Bedeutung. Sie werden hier als gegeben angenommen, unabhängig vom Grad der Vorverarbeitung. Somit sind rohe, ungefilterte Sensordaten ebenso valide wie höhere Perzepte; zum Beispiel die horizontale Bildposition eines von der Bildverarbeitung erkannten Objektes. Die sensorische Information, welche zum aktuellen Zeitpunkt $t \in \mathbb{N}$ verfügbar ist, wird mit dem Sensorvektor $\mathbf{x}(t) \in S \subseteq \mathbb{R}^D$ bezeichnet, wobei D die Anzahl der verschiedenen Sensorkanäle x_i mit $i = 1 \dots D$ ist. S ist der sensorische Zustandsraum, d. h. die Menge aller vom System potentiell annehmbaren Zustände. Die Zeit verstreicht dabei in äquidistanten Schritten. Die Größe dieser zeitlichen Abstände wird durch die verwendete Recheneinheit vorgegeben und kann beispielsweise 10 ms pro Zeitschritt betragen. Der Sensorvektor des Systems wird also 100 Mal pro Sekunde aktualisiert.

Die andere Schnittstelle zur *Außenwelt* sind die Aktuatoren. Für das Modell sind auch sie nicht konkret vorgegeben. Üblicherweise sind das z. B. mechanische Aktuatoren, um den Zustand des Körpers oder der Umwelt zu verändern. Genauso gut könnten es aber Lampen oder auch Lautsprecher sein. Der Zweck steht hierbei nicht im Vordergrund, da das erlernte Verhalten unabhängig von einer konkreten Aufgabe sein soll. Dennoch werden zwei Anforderungen an die Aktuatorik gestellt: Erstens, die

Ausübung einer motorischen Handlung soll auch in einer für das Individuum wahrnehmbaren Umweltreaktion münden. Ein Aktuator ergibt nur dann einen Sinn, wenn er auch etwas bewirken kann. Das heißt nicht, dass *jede Art der Ausübung* auch zwingend eine Reaktion erwirken muss. Neutrale Aktionen sind dabei durchaus zugelassen. Zweitens sollen die Aktuatoren keine Aktionen ausführen können, welche die Funktion des Systems negativ beeinflussen. Das Individuum darf sich nicht selbst beschädigen können. Für eine Untersuchung unter Ausschluss extrinsischer Motive (vgl. Abschnitt 2.4) ist diese Forderung notwendig, da dem Individuum explizit keine Rückkopplung über möglicherweise schädliche Aktionen gegeben wird. Ansonsten sei der Körper und sein Funktionsumfang, welcher er dem Individuum zur Verfügung stellt, hier ebenfalls als beliebig und gegeben anzunehmen.

Bei natürlichen Lebewesen wächst der Körper in gleicher Weise wie auch das in ihm beherbergte informationsverarbeitende System heranwächst. Das körperliche System unterliegt also im Allgemeinen gewissen Veränderungen, mit welchen das lernende System umgehen können muss. In der Robotik geht man zur Zeit noch allgemein von einer von Beginn an fertigen Morphologie aus. Nichtsdestoweniger unterliegt auch diese Veränderungen, wie z. B. Verschleiß. Das Modell geht also nicht von einer konstanten Art und Weise der Ausübung der Aktionen aus. Gelegentliche Veränderungen oder schleichender Verschleiß sind als Teil des Körpers mitinbegriffen.

Die Schnittstelle für das lernende System ist der Motorvektor $\mathbf{m}(t) \in \mathbb{R}^{\Lambda}$, wobei Λ die Anzahl der verschiedenen motorischen Subsysteme ist. Beispielsweise gibt $m_j(t)$ das Steuersignal für das j-te Gelenk des Individuums vor, welches dann von einem im Gelenk angebrachten Servomotor ausgeführt wird. Um zuverlässig die Konsequenzen des eigenen Handelns abschätzen zu können, ist es offenbar von Vorteil, die generierten Motorsignale als zusätzliche sensorische Eingabe zu verwenden. Die sensorische Information $\mathbf{x}(t)$ kann also bereits den im letzten Zeitschritt generierten Motorvektor enthalten. Zu den Sensorinformationen zählen demnach auch intern generierte Signale. Ohne Beschränkung der Allgemeinheit seien sowohl der Sensorvektor, als auch der Motorvektor auf das Intervall $[-1, +1]$ beschränkt. Das ist eine übliche Konvention und hat vor allem praktische Vorzüge bei der Verwendung neuronaler Methoden.

Die Sinnesorgane biologischer Lebewesen, genau wie die Sensoren eines Roboters, sind in der Lage feine Änderungen der jeweiligen Umwelteigenschaft zu detektieren. Doch obwohl die Sinneswahrnehmungen kontinuierlich sind, ist offenbar die grobe Diskretisierung kontinuierlicher Eigenschaften ein probates Mittel, um die Komplexität der Verarbeitung zu reduzieren. So unterteilt der Mensch Farben in vergleichsweise groben Kategorien wie rot, gelb und grün oder assoziiert diese mit Alltagsentitäten wie Orangen oder Oliven. Die Auflösung dieser Einteilung ist dabei auch an den Kontext gebunden. Die gefühlte Temperatur von Wasser teilt man gerne in eiskalt, kalt, lauwarm, warm, heiß und kochend ein; wohingegen oft für eine Gefahreneinschätzung der Herdplatte die Unterscheidung in heiß und kalt ausreichend ist.

Lebewesen sind offensichtlich in der Lage, aus kontinuierlich fließenden sensorischen Informationen diskrete interne Zustände abzuleiten. Dies wird bei lernenden Individuen zu einem großen Teil durch erlebte Situationen bestimmt. Die frühkindlichen Erfahrungen prägen in entscheidender Weise wie sensorische Information vom System verarbeitet wird. Dabei ist nicht in jedem Fall klar, wie viele dieser Umweltrepräsentationen schon durch Evolution erworben wurden und bereits von Beginn an verfügbar

sind. Für das hier beschriebene Modell wird dieser Umstand radikalisiert und somit für den Beginn des Lernprozesses eine *Tabula-Rasa*-Situation angenommen. Das Individuum ist also zunächst nichts weiter als ein unbeschriebenes Blatt.

4.2 Aufteilung des Zustandsraums

Zu Beginn sind die in das System gelangenden Sensorinformationen zeitdiskrete Werte ohne Bedeutung für das Individuum. Die Aufgabe des Systems ist es nun, die eintreffenden Informationen angemessen *abzuspeichern* und einzuordnen. Die Ausgangssituation ist dabei mit dem Aufbau der eigenen Wohnzimmerbibliothek vergleichbar. Die ersten fünf Bücher stellt man wohl eher unsortiert in das Regal. Kommen schrittweise neue Bücher hinzu, verliert man womöglich bald den Überblick und man beginnt grobe Kategorien zu bilden und das Regal einzuteilen. Dabei kann diese Kategoriebildung durchaus sehr unausgeglichen sein. Es mag vorkommen, dass die Kategorie »Wissenschaft & Technik« schon dreißig Bücher umfasst, von denen 29 etwas mit Informatik oder Physik zu tun haben, während in der Kategorie »Belletristik« zwei ungelesene geschenkte Exemplare verweilen. Die konkrete Ausgestaltung der Kategoriebildung ist demnach von der Beschaffenheit und Quantität der zu kategorisierenden Information abhängig.

Wieder in Bezug auf das lernende System bedeutet das nun, dass eintreffende Sensorinformationen zu Beginn nur in einer Kategorie aufgenommen und gespeichert werden. Wird diese Kategorie zu groß, so findet eine Unterteilung statt. Dieser Prozess setzt sich weiter fort. Aufgrund beschränkter Speicherressourcen, kann das System nicht alle Sensorinformation lebenslang speichern. Selbst unter der Annahme, dass ein genügend großer Speicher vorhanden wäre, würde die Zeit die es braucht, um die vorhandene Information zu verwalten, ebenfalls immerzu anwachsen und das Individuum könnte nicht mehr angemessen auf Umweltreize reagieren. Im Wesentlichen ist es auch gar nicht notwendig alle Informationen aufzubewahren. Prinzipiell würde es reichen, sich *Neues* oder *Interessantes* zu merken, wobei man nur entscheiden kann was neu ist, wenn man bereits weiß was alt ist – scheinbar ein Dilemma.

Um die eintreffende Sensorinformation angemessen zu verarbeiten, kann man auf das bereits in Abschnitt 3.2.3 vorgestellte Wettbewerbslernen zurückgreifen. Jeder entstehenden Kategorie ordnet man dazu eine eigene Lernmaschine zu. Diese wird als *Experte* bezeichnet. Alle Experten $n = 1 \dots N$ machen nun ihre Vorhersage $\hat{\mathbf{x}}_n(t+1)$ über die zu erwartenden Sensorwerte $\mathbf{x}(t+1)$. Dafür können sie Gebrauch von gespeicherter Information vergangener Zeitschritte machen. Die Vorhersagen werden nun im nächsten Zeitschritt mit den tatsächlichen Werten verglichen und dabei der Fehler

$$E_n(t) = \|\mathbf{x}(t) - \hat{\mathbf{x}}_n(t)\|^2 \tag{4.1}$$

als quadrierte euklidische Norm aus der Differenz der wirklichen Sensorwerte zur Vorhersage errechnet. Die beste Schätzung zeichnet sich dabei durch den geringsten Fehler aus und der Experte

$$s_1 = \arg\min_n E_n(t) \tag{4.2}$$

mit der besten Schätzung wird zum Gewinner ernannt. Dieser übernimmt nun den richtig geschätzten Sensorwert in seine Kategorie. Dazu wird der Experte auf den neuen

Sensorwert trainiert, um zukünftig noch bessere Vorhersagen für ähnliche sensorische Information zu machen. Wie genau ein Experte aufgebaut ist und lernt, erläutert Abschnitt 5.1. Der Sensorwert wird danach verworfen und *nicht gespeichert*. Die Information steckt danach implizit in der verbesserten Vorhersagefähigkeit des Experten. Alle anderen Experten werden nicht angepasst. Das führt alsbald zu einer Spezialisierung, bei der jeder Experte eine Nische besetzt und für einen *bestimmten sensomotorischen Kontext zuständig* ist. Der Zustand des Systems wird also durch die jeweiligen Experten angezeigt. Ein System, das eintreffende Sensorwerte mithilfe seiner Experten in N Kategorien einteilen kann, ist also bereits in der Lage auch N verschiedene sensorische Situationen zu unterscheiden und vor allem wiederzuerkennen, wenn sie erneut auftreten. Ein solches System bezeichnet man als *Multi-Experten-Architektur* oder auch als *Competing Experts*.

4.3 Verwaltung der Experten

Die Aufteilung einer Kategorie heißt, einen Experten zu klonen, weil eine sinnvolle Teilung einer Lernmaschine in der Regel schwierig oder gar nicht möglich ist. Damit sich fortan beide Experten auf unterschiedliche Bereiche spezialisieren, muss die vorhandene Symmetrie gebrochen werden. Dazu werden beim Klonen der Experten kleine, aber für die korrekte Funktion unbedeutende Fehler gemacht. Der neu erzeugte Experte wird zudem noch ein wenig in Richtung des aktuellen Sensorvektors trainiert. Damit ist die Genese abgeschlossen.

Bleibt zu klären, wann oder nach welchen Kriterien eine Teilung vollzogen werden sollte. Dazu könnte man die Anzahl der verwendeten Sensorwerte zählen und nach einer festgelegten Höhe die Kategorie teilen. Allerdings kommt es vor, dass eine Zeit lang kaum Varianz in den Sensorwerten ist und das Potential eines Experten damit unnötig verschenkt wird. Es sollte also ein Maß gefunden werden, was die Entwicklung des Experten berücksichtigt. Ein Experte sollte dabei nach einer bestimmten Zeit ausgelernt haben und das Feld anderen Experten überlassen, wenn er genügend gelernt hat. Dazu kann man für jeden neuen Experten ein Lernkontingent festlegen, welches irgendwann aufgebraucht wird. Wenn das der Fall ist, wird die Kategorie geteilt. Der Unterschied zur Abzähl-Methode ist, dass das Kontingent nur verringert wird, wenn der Experte auch wirklich etwas gelernt hat. Dafür muss ermittelt werden, wieviel Anpassung wirklich notwendig war, um die letzte Sensorinformation zu verarbeiten. Treten dabei oft gleichartige Sensorwerte auf, muss nicht viel für die Anpassung des Experten getan werden und sein Lernkontingent wird demnach auch nicht aufgebraucht.

Auch einer fortschreitenden Kategoriebildung muss eine obere Schranke gesetzt werden, damit das System nicht über alle Maße wächst. Dabei stellt sich die Frage, was passiert, wenn die Anzahl N der Experten die Schranke N_{max} erreicht hat. Eine Lösung dafür konnte in [13] gefunden werden (vgl. dazu Abschnitt 3.2.3). Für alle Experten wird ein Maß für deren Nützlichkeit im Gesamtsystem ermittelt. Möglicherweise hat nicht jede Genese auch zu nützlichen Spezialisierungen geführt. Außerdem muss beim Erreichen der oberen Grenze wieder Platz für neue Informationen geschaffen werden. Die Nützlichkeit wird anhand der Performanz der Experten bemessen. Gleichung (3.16) definiert die Nützlichkeit des Gewinnerexperten als die Differenz der Vorhersagefeh-

ler von Vize und Gewinner. Damit ist ein Experte umso nützlicher, je schlechter der zweitbeste Experte die Vorhersage bewältigt hat. Somit kann der Experte mit der geringsten Nützlichkeit von seiner Aufgabe entbunden und an anderer, erwartungsvoller Stelle wieder eingesetzt werden.

4.4 Exploration und Evaluation des Lernfortschritts

Im Hinblick auf die Frage, wie neue Information in das System gelangt, kann man vorerst davon ausgehen, dass für ein ausgewogenes Lernen ausreichend Abwechslung vorhanden sein muss. Verharrt das Individuum immerzu in demselben sensomotorischen Kontext, führt das zwangsläufig zu einer unausgewogenen Kategoriebildung. Beispielsweise kann das System zwar bereits zwanzig verschiedene Grüntöne unterscheiden, hat aber noch nie ein Blau wahrgenommen und das, obwohl die Farbe Blau sich in unmittelbarer und erreichbarer Nähe befindet. Es ist demnach für die ausgewogene Entwicklung der Wahrnehmung enorm wichtig, dass ein Individuum abwechslungsreicher, sensorischer Information ausgesetzt ist. Es muss also *explorieren*. Die treibende Kraft explorativen Verhaltens ist dabei der Zufall.

Vorerst befindet sich das Individuum in der Rolle des Beobachters. Es wird durch anfänglich zufällige Bewegungen stetig neuen Situationen ausgesetzt, welche es nach und nach besser einordnen kann. Anfangs sind die Voraussagen der Experten nicht besonders gut. Im Laufe der Entwicklung lernen die einzelnen Experten immer mehr dazu und die Vorhersagen werden zunehmend besser. Durch die Spezialisierung und Nischenbesetzung verbessert sich dadurch auch die Gesamtvorhersage, welche sich aus den Einzelvorhersagen der jeweiligen Gewinner ergibt. Möchte man nun messen, wie und ob etwas gelernt wird, kann man dazu den Fehler der Vorhersage verwenden. Macht das System Fortschritte beim Lernen, so erwartet man, dass der Fehler im Mittel über die Zeit geringer wird. Im »Mittel« daher, weil nicht jede Anpassung auch dazu führt, dass kommende Vorhersagen zwangsläufig besser werden. Sensordaten sind im Allgemeinen verrauscht und können auch fehlerhafte Informationen enthalten. Daher kann zeitweilig eine Adaption auch zu schlechteren Vorhersagen führen.

Der *erzielte Lernfortschritt* kann nun als das Absinken des Vorhersagefehlers formuliert werden. Wird der Fehler geringer, so wurde offenbar dazu gelernt. Steigt der Fehler hingegen wieder an, gibt es möglicherweise Probleme die Sensorinformation richtig zu verarbeiten. Die eigentliche Höhe des Fehlers ist dabei weniger von Interesse. Ein zeitweilig geringer Fehler sagt im Allgemeinen noch nichts über den Lernfortschritt aus. Es bedeutet nur, dass der aktuelle Zustand präzise vorhergesagt werden kann. Das kann aber auch darin begründet sein, dass gerade nichts Spannendes passiert, was gelernt werden könnte. Solche Situationen würde man üblicherweise als *langweilig* oder *unterfordernd* bezeichnen. Genauso gut ist es denkbar, dass sich sensorische Situationen ergeben, die so komplex sind, dass sie *prinzipiell* nicht vom System erlernt werden können und egal wie lange gelernt wird, der Fehler dabei nicht geringer wird. Die interessante Größe ist also vielmehr die zeitliche Änderung des Vorhersagefehlers. Formell sei somit der Lernfortschritt

$$L(t) = -\frac{\mathrm{d}E_{s_1}(t)}{\mathrm{d}t} \tag{4.3}$$

definiert als die negative Ableitung des Fehlers des Gewinner-Experten s_1 nach der

Zeit. Der Fehler ist im Wesentlichen eine Funktion der Sensorwerte, demnach wird darin enthaltenes Rauschen prinzipiell bei einer Ableitung verstärkt. Dem kann auf unterschiedliche Weise begegnet werden. Wie eine rauscharme diskrete Ableitung durchgeführt wird, ist ausführlich in Kapitel 6 beschrieben.

4.5 Zustandsbewertung und Auswahl motorischer Aktionen

Wie kann man nun diesen Lernfortschritt steigern? Schließlich soll das Individuum so viel wie möglich über sich und seine Umwelt in Erfahrung bringen, bzw. sich kontinuierlich an Veränderungen anpassen können. Bisher übt das Individuum ausschließlich zufällige Aktionen aus. Das Verhalten ist also bisher noch wenig intentional. Es kann möglicherweise seinen Lernfortschritt erhöhen, wenn es gezielt die Aktionen ausübt, welche bereits zu Lernfortschritt geführt haben. Dazu muss es sich die entsprechenden Aktionen merken, abwarten und genau dann ausführen, wenn es sich wieder in derselben Situation befindet. Es muss also jeder Experte, welcher einem bestimmten sensomotorischen Kontext zugeordnet ist, eine Liste mit den zur Auswahl stehenden Aktionen haben und gewissermaßen Protokoll führen, welche Aktionen sich bewährt haben. Auf der Basis seiner Bewertungen wählt nun der Gewinner die vielversprechendste Aktion aus und beobachtet das Resultat zum nächsten Zeitschritt. Mit der Anzahl der Versuche kann er seine Schätzung über die fruchtbarste Aktion verbessern. Somit ergibt sich intentionales Verhalten. Eine Handlung unter präzisen Zielvorstellungen hat nach [9] ein Motiv. Die Motivation ist dabei die Maximierung des Lernfortschritts. Das System belohnt sich also selbst für erfolgreiches und fortschreitendes Lernen, ist also intrinsisch motiviert, wobei die Belohnung äquivalent zum erreichten Lernfortschritt ist (vgl. dazu Abschnitt 2.4). Die Belohnung, welche das System erhält, ist definiert als

$$r(t) = L(t), \tag{4.4}$$

also der gemessene Lernfortschritt zum aktuellen Zeitpunkt. Die erhaltene Belohnung wird für die Bewertung von Zuständen und Aktionen gebraucht. Dazu wird ein Verfahren aus dem Bereich des *bestärkenden Lernens* verwendet, welches in Kapitel 7 beschrieben wird.

Interessant ist die Frage was passiert, wenn der Lernprozess langsam stagniert. Genauer: Wenn durch fortschreitendes Lernen der Fehler sich seinem Minimum nähert, wird mit der Zeit die zu erwartende Belohnung ebenfalls geringer. In diesem Fall muss das Individuum wieder explorieren und durch zufällige Aktionen neue Situationen auffinden, in denen es etwas lernen kann. Darin kann es nun wieder verweilen und lernen. So setzt sich dieser Prozess fort. Das intentionale Handeln kann gewissermaßen als Gegenkraft zur Exploration verstanden werden. Demnach ist die Aktionsauswahl ein Wechselspiel aus rein zufälligen und mit der Erwartung auf Lernfortschritt ausgewählten Aktionen.

Die Argumentation der letzten Absätze stützt sich auf eine ebenfalls diskrete und begrenzte Anzahl verschiedener motorischer Aktionen. Daher muss noch abschließend geklärt werden, wie aus einer Liste M diskreter Aktionen, der eingangs definierte reelle Motorvektor $\mathbf{m}(t)$ gebildet wird. Pauschal kann das Problem an dieser Stelle nicht

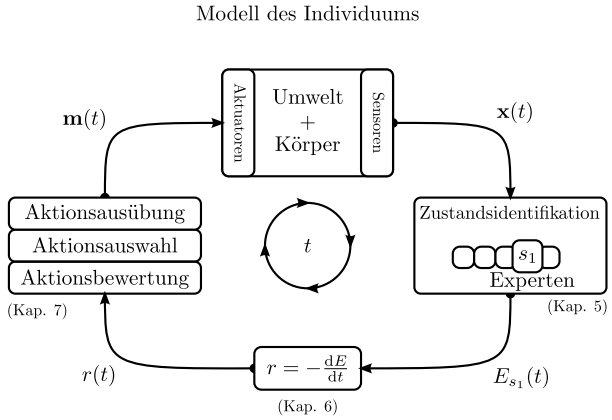

Abbildung 4.1: Schematische Darstellung des Modells für ein abstraktes Individuum. Die über die Sensorik wahrgenommene Umweltinformation wird durch die Experten eingeordnet und versucht die nächsten Sensorwerte vorherzusagen. Der Gewinner-Experte s_1 mit der besten Vorhersage identifiziert den aktuellen Zustand. Die negative Ableitung des Vorhersagefehlers, d. h. der Lernfortschritt, wird als Belohnung interpretiert und zur Bewertung der Zustände und Aktionen verwendet. Auf der Grundlage dieser Bewertungen wird die nächste motorische Aktion ausgewählt und über die Aktorik ausgeführt.

gelöst werden. Es muss in Abhängigkeit der spezifischen Aktuatoren des Individuums betrachtet werden. Für die in dieser Arbeit untersuchten Morphologien mit zwei motorischen Freiheitsgraden, beschreibt der Abschnitt 7.3.2 ein neuronales Motormodul, welches den dementsprechend zweidimensionalen Motorvektor in M verschiedene Winkel aufteilt. Beispielsweise könnte man für ein Individuum mit einem Lämpchen als Aktuator die diskreten Aktionen in *an* und *aus* oder ggf. mit Zwischenstufen einteilen. Auch relative Aktionen wie *aufhellen* und *abdunkeln* bieten sich an.

Abschließend ist das vollständige Modell in der Übersichtsgrafik 4.1 abgebildet. Die folgenden Kapitel untersuchen nun systematisch mögliche Realisierungen der einzelnen Module (im Uhrzeigersinn), beginnend mit der Wahrnehmung und Zustandsidentifikation in Abschnitt 5.1. Darauf folgt die Darstellung eines modifizierten wachsenden neuronalen Gases zur Verwaltung der Multi-Experten-Archtitektur in Abschnitt 5.2. Kapitel 6 beginnt mit dem Entwurf eines Filters für die robuste Ableitung der diskreten Fehlersignale und fährt mit der Analyse der resultierenden Lernsignale fort. Kapitel 7 beschreibt ein bewährtes Verfahren zur Bewertung von Zuständen und Aktionen, anhand der eingeholten Belohnung, wobei die Abschnitte 7.2 und 7.3 beschreiben, wie die bewerteten Aktionen ausgewählt bzw. ausgeübt werden.

5 Zustandsidentifikation

Die Wahrnehmung des Individuums wird wie in Kapitel 4 beschrieben als Multi-Experten-Architektur modelliert. Dazu muss geklärt werden, wie ein einzelner Experte aufgebaut ist und wie diese im Zusammenspiel (Wettbewerbslernen) funktionieren. Das Kapitel zur Zustandsidentifikation ist dazu wie folgt aufgebaut: Der erste Abschnitt beschreibt den Aufbau eines einzelnen Expertenmoduls und seine Funktionsweise. Der zweite Abschnitt erläutert darauf die Verwaltung der einzelnen Experten und wie mit deren Hilfe der Zustand des Systems identifiziert wird. Weiterhin wird gezeigt, wie auf Basis eines modifizierten Wachsenden Neuronalen Gases neue Experten eingefügt und ggf. als unnütz identifizierte Experten wieder neu eingeordnet werden.

5.1 Prädiktionsmodule

Der folgende Abschnitt beschreibt die Auswahl, den Aufbau und den Funktionstest von Prädiktormodulen für die Vorhersage zukünftiger sensorischer Informationen. Der Abschnitt ist dabei wie folgt unterteilt: Zuerst werden für die Auswahl wichtige Vorüberlegungen getätigt. Darauf folgend wird ausgehend von einem einfachen Ansatz das Modell zunehmend komplexer und alternative Ansätze aufgezeigt. Dabei werden die Eigenschaften der vorgestellten Verfahren erläutert und die Vor- und Nachteile abgewogen. Ein einfaches Lernszenario erläutert dabei die Funktionsweise. Der Abschnitt endet mit dem Fazit und begründet die schlussendlich getroffene Auswahl.

5.1.1 Vorüberlegungen und Auswahlkriterien

Prinzipiell kommen mehrere Verfahrensklassen zur Zeitreihenvorhersage oder Systemidentifikation in Frage. In dieser Arbeit werden speziell neuronale oder dem ähnliche Architekturen untersucht. Als Trainingsverfahren für die Prädiktionsmodule wird die in Abschnitt 3.2.2 vorgestellte *Backpropagation-Lernregel*, d. h. ein verallgemeinerter *Least-Mean-Squares-Algorithmus* verwendet. Die folgenden Strukturen sind demnach entweder reine *Feed-Forward-Netze* oder haben nur lokale rekurrente Verbindungen, welche entweder gar nicht oder mit einer speziellen Lernregel langsam angepasst werden. Vollständig rekurrente Netze als universelles Modell sind sehr mächtig und in der Lage überaus komplexe Dynamiken bis hin zu chaotischen Zeitreihen abzubilden. Das Training gestaltet sich aber mangels effizienter Verfahren sehr kostenintensiv. In Abschnitt 3.2.2 sind die problematischen Eigenschaften solcher Lernregeln und der Grund für ihren Ausschluss erläutert.

Zeit und Gedächtnis

Die Art und Weise wie die Zeit innerhalb der Module verarbeitet wird ist ein wichtiges Kriterium für die Auswahl der Prädiktionsmodule. Die Modelle müssen in der Lage sein, den zeitlichen Verlauf und die zeitlichen Abhängigkeiten des Systems zu erfassen und abzubilden [10]. Daraus folgt, dass ein solches Modell ein Gedächtnis braucht. Die Implementation eines solchen Speichers kann zum Beispiel durch eine zeitliche Einbettung der bereits bestehenden Sensordaten erfolgen, d. h. das Modul kann die im Speicher abgelegten zeitverzögerten Kopien bereits veralteter Sensordaten für eine Prädiktion in die Zukunft verwenden. Eine solche zeitliche Einbettung mittels einer anzapfbaren Verzögerungskette (engl. *tapped delay line*) gilt zwar bisher als biologisch unplausibel, ist aber vergleichsweise einfach zu implementieren und robust zu trainieren. Durch das Ablegen exakt zeitlich getakteter Sensordaten werden solche oder ähnlich Verfahren als explizite Zeiteinbettung bezeichnet. Die explizite Zeiteinbettung entspricht somit einer örtlichen Repräsentation der Zeit als zusätzliche Eingaben. Dabei kommt die Frage auf, wie viele zeitlich verzögerte Kopien man dafür vorhalten muss. Die *implizite* Einbettung der Zeit geschieht, im Gegensatz dazu, wenn die Propagierung der Daten im verarbeitenden System selbst verzögert wird und beispielsweise durch rekurrente Verbindungen dazu führt, dass bereits vergangene Sensordaten noch die Vorhersage beeinflussen. Innerhalb dieses Kapitels werden beide Verfahren vorgestellt und die Vor- und Nachteile benannt.

Vorhersageaufgaben

Als weitere Vorüberlegung für den Entwurf eines Prädiktionsmoduls gilt es die Vorhersageaufgabe für das Modul festzulegen. Die typischen Aufgaben eines Prädiktormoduls sind *Zeitreihenvorhersage* und *Systemidentifikation*. Die Aufgaben sind sich sehr ähnlich und verwenden dieselbe lernende Struktur. Der eigentliche Algorithmus unterscheidet sich nur darin welche Daten die Grundlage der Voraussage sind.

Bei der Zeitreihenvorhersage wird dem Prädiktor die bekannte Vergangenheit der Zeitreihe bereitgestellt und erwartet, dass er die künftigen Werte des unbekannten Prozesses korrekt voraussagt. Die Bereitstellung der Vergangenheit geschieht demnach entweder durch eine begrenzte Anzahl explizit zeitverzögerter Kopien vergangener Werte der Zeitreihe oder die Information wird intern mithilfe rekurrenter Verbindungen gespeichert. Trainiert wird der Prädiktor durch den Vergleich der aktuellen Schätzung mit dem zwangsläufig eintreffenden neuen Wert der Zeitreihe. Das Ergebnis wird im Allgemeinen umso unpräziser, je weiter dabei in die Zukunft geschaut wird. Koppelt man ein trainiertes Modul von der echten Zeitreihe ab und füttert es mit seinen eigenen Voraussagen, so laufen die Trajektorien in der Regel nach kurzer Zeit auseinander. Wichtig dabei ist, dass der prädizierte Verlauf nicht zu schnell divergiert. Als Maß für die Stärke des *Auseinanderlaufens* zweier Prozesse (echte Zeitreihe und geschätzte Zeitreihe) bietet sich der Ljapunov-Exponent an.

Eine weitere beliebte Aufgabe für Prädiktoren ist die Systemidentifikation. Dazu gibt man ein und dasselbe Zufallssignal (z. B. ein gleichverteiltes Rauschen) einmal durch das unbekannte System und einmal durch den Prädiktor. Trainiert wird wieder durch den Vergleich zwischen Systemverhalten und Schätzung. Bei einem erfolgreichen

Training hat der Prädiktor die unbekannten Eigenschaften des zu identifizierenden Systems erlernt und reagiert auf gegebene Eingabesignale dem System sehr ähnlich, d. h. er kann das Verhalten des Systems approximieren. Auf diese Weise lässt sich beispielsweise die Charakteristik eines unbekannten Filters erlernen, wie innerhalb dieses Kapitels demonstriert wird.

5.1.2 Allgemeiner Aufbau eines Prädiktionsmoduls

Die Grundaufgabe eines Prädiktionsmoduls besteht in der Vorhersage der Sensorwerte. Dazu stehen dem Modul zu jedem Zeitschritt t die D verschiedenen Sensorwerte

$$\mathbf{x}(t) = (x_1(t), x_2(t), \ldots, x_D(t))^T \tag{5.1}$$

zur Verfügung. Benötigt das Modul für die Vorhersage $\hat{\mathbf{x}}(t+1)$ zusätzlich die Vergangenheit der Sensorwerte, so muss es einen entsprechenden internen Speicher besitzen. Treffen zum nächsten Zeitpunkt $t+1$ die neuen Sensordaten $\mathbf{x}(t+1)$ ein, so wird die Abweichung

$$\mathbf{e}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t) \tag{5.2}$$

der Vorhersage zu den wirklichen Daten ermittelt. Daraus errechnet sich der Gesamtfehler

$$E(t) = \|\mathbf{x}(t) - \hat{\mathbf{x}}(t)\|^2 = \sum_i e_i(t)^2. \tag{5.3}$$

Dieser wird nun verwendet, um über das in Abschnitt 3.2.2 vorgestellte Gradientenverfahren den noch zu spezifizierenden Parametersatz $\mathbf{W}$ des Moduls anzupassen. Dazu wird die Stärke der Adaption durch die Lernrate $\eta \in \mathbb{R},\ 0 < \eta \ll 1$ reguliert. Ein geringer Fehler in der Vorhersage wird als hohe Güte interpretiert. In Abschnitt 5.2 wird dieses Signal verwendet, um zu entscheiden, welches Prädiktionsmodul die beste Güte hat, d. h. die wenigsten Fehler macht. Abbildung 5.1 zeigt zusammenfassend den schematischen Aufbau eines allgemeinen Prädiktormoduls.

Allgemeiner Aufbau eines Prädiktormoduls

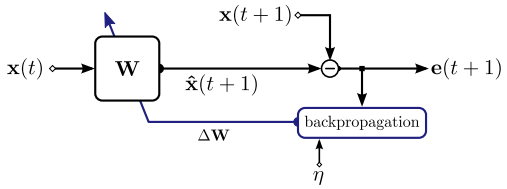

Abbildung 5.1: Allgemeiner Aufbau eines Prädiktormoduls.

5.1.3 Vorhersage durch Mittelwertschätzung

Eine einfache Form der Schätzung zukünftiger Sensorwerte kann über einen gleitenden Mittelwert erfolgen. Das erfordert nichts weiter als die Anpassung eines einzigen

Gewichts b_j (d. h. dem Biasgewicht) pro Dimension j der Vorhersage $\hat{\mathbf{x}}(t)$. Wenn sich die Sensordaten nur langsam ändern liefert

$$\hat{x}_j(t) = b_j \tag{5.4}$$

eine zuverlässige Schätzung $\hat{x}_j(t)$ und es erfordert lediglich den entsprechenden aktuellen Sensorwert $x_j(t)$ zur Anpassung der Schätzung. Die einfache Lernregel

$$\Delta b_j = \eta \left(x_j(t) - \hat{x}_j(t) \right) \tag{5.5}$$

passt die Schätzung an und wenn die Lernrate η hinreichend klein eingestellt ist, so verfolgt $\hat{\mathbf{x}}(t)$ den Mittelwert des zeitlichen Verlaufs der Sensorwerte $\mathbf{x}$. Diese Form der Lernregel wurde bereits in Abschnitt 3.2.3 vorgestellt und findet seine Verwendung zur Anpassung der Gewichte eines wachsenden neuronalen Gases.

Für langsam veränderliche Signale mit geringer Dynamik ist diese Form der Schätzung ein probates Mittel um kostengünstig verschiedene statische Zustände voneinander zu trennen. Die Methode hat allerdings keine Möglichkeit verschiedene dynamische Zustände voneinander zu trennen, wenn diese denselben Mittelwert ergeben. Um das zu illustrieren stelle man sich ein Pendel vor, an dem ein Winkelgeber dessen aktuelle Auslenkung um die Nullposition misst. Gibt nun der Sensor beispielsweise eine Auslenkung von Null an, kann nicht unterschieden werden, ob sich das Pendel in der Ruhelage befindet oder gerade durch diese hindurch schwingt. Der Positionssensor deckt bei diesem System nicht alle Dimensionen des Zustandsraums ab. Er kann nur den Ort bestimmen; die Geschwindigkeit aber fehlt. Das Wissen über die Geschwindigkeit des Pendels ist jedoch notwendig, damit der Zustand eindeutig bestimmbar ist. Wie bereits in Abschnitt 2.2 diskutiert, deckt die sensorische Ausstattung in den seltensten Fällen den vollständigen Zustand ab. Daher muss die Historie der Sensordaten verwendet werden, um Eigenschaften wie die Änderung der Position oder die Geschwindigkeit der Änderung indirekt zu bestimmen. Wie eingangs erwähnt steht für die Prädiktion der gesamte Sensorvektor $\mathbf{x}(t)$ zur Verfügung. Die Schätzung kann noch besser werden, wenn für die Vorhersage auch die Informationen der anderen Sensoren Verwendung finden.

5.1.4 FIR-Prädiktor

Der hier vorgestellte FIR-Prädiktor verwendet alle zur Verfügung stehenden Sensordaten $\mathbf{x}(t)$ und führt eine Expansion der Daten in der zeitlichen Dimension durch. Von jedem Sensordatum $x_j(t)$ werden $K \in \mathbb{N}$ Werte der Trajektorie bereitgehalten. Der vollständige Eingabevektor für den Prädiktor ist somit $\tilde{\mathbf{x}}(t) \in \mathbb{R}^{DK+1}$ mit

$$\tilde{\mathbf{x}}(t) = \left(\mathbf{x}(t), \mathbf{x}(t-1), \ldots, \mathbf{x}(t-K+1), 1 \right)^T \tag{5.6}$$

also ein einziger Spaltenvektor aller verfügbaren Sensorwerte (5.1) inklusive der $K-1$ zeitverzögerten Kopien und dem Bias. Die Vorhersage $\hat{\mathbf{x}}(t+1)$ der zukünftigen Sensorwerte ergibt sich nun als

$$\hat{\mathbf{x}}_{t+1} = \tanh\left(\mathbf{W} \tilde{\mathbf{x}}_t \right), \tag{5.7}$$

d. h. aus der Multiplikation des expandierten Eingabevektors mit der Gewichtsmatrix $\mathbf{W} \in \mathbb{R}^{D \times (KD+1)}$ und der Begrenzung durch den Tangens Hyperbolicus. Gedanklich

kann man alle zu einem Kanal zusammengehörigen Sensorwerte als die Eingabe von $x_j(t)$ über eine spezielle Form der Synapse verstehen. Im Fall der zeitverzögerten Kopien entspricht das exakt einer Filterung mit einem sogenannten FIR-Filter, einem Filter mit endlicher Impulsantwort (engl. *finite impulse response filter*). Eine solche

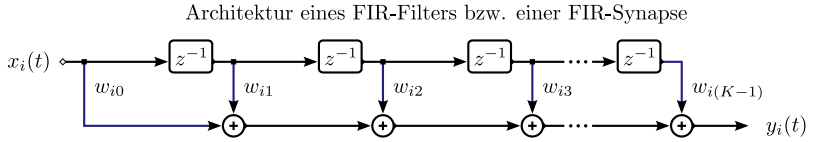

Abbildung 5.2: Allgemeines Modell für ein Filter mit endlicher Impulsantwort.

Synapse hat demnach Filtereigenschaften, welche durch Anpassung der Gewichte, d. h. der Filterkoeffizienten, erlernt werden können. Daher wird hier zur Vereinfachung die FIR-Synapse [1] eingeführt. Abbildung 5.2 zeigt den schematischen Aufbau eines allgemeinen FIR-Filters und somit auch der FIR-Synapse. Der Ausgang des Filters (bzw. der Synapse) berechnet sich durch

$$y_i(t) = \sum_{k=0}^{K-1} w_{ik} x_i(t-k), \tag{5.8}$$

wobei w_{ik} die *lernbaren Filterkoeffizienten* sind und $x_i(t)$ die Eingabe des i-ten Sensorkanals. Der Aufbau des FIR-Prädiktors für *eine einzelne Komponente* $\hat{x}_j(t+1)$ der Vorhersage ist in Abbildung 5.3 gezeigt. Für die Berechnung des ganzen Vorhersagevektors $\hat{\mathbf{x}}(t+1)$ werden demnach D Neuronen und $KD^2 + D$ Synapsen, oder D^2 FIR-Synapsen und D Biassynapsen benötigt. Zur Reduktion des Rechenaufwands kann die Anzahl der Vorhersagen reduziert werden, indem nur ein Teil der Komponenten von $\hat{\mathbf{x}}(t+1)$ berechnet wird. Dazu werden nur die aussagekräftigsten Sensorkanäle vorhergesagt. Welche das im Detail sind muss in Abhängigkeit der Anwendung ausgemacht werden.

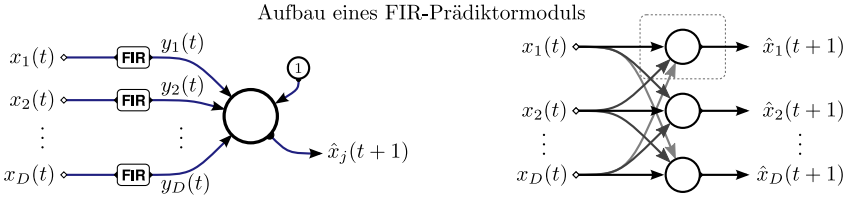

Abbildung 5.3: Schematischer Aufbau des FIR-Prädiktormoduls für eine Komponente $\hat{x}_j(t+1)$ (links) und für den gesamten Vorhersagevektor $\hat{\mathbf{x}}(t+1)$ (rechts).

Funktionstest

Um die korrekte Funktion zu überprüfen wird eine Aufgabe zur Systemidentifikation gewählt. Dazu wird ein normalverteiltes Rauschsignal durch ein bekanntes, zu identifizierendes FIR-Tiefpassfilter mit den drei konstanten Koeffizienten $c_0 = 0{,}25$, $c_1 = 0{,}5$

und $c_2 = 0{,}25$ gegeben. Der Ausgang dient dem lernenden Filter als Trainingssignal. Das lernende Filter beginnt mit vier zufällig aus dem Intervall $[-0{,}1; 0{,}1]$ initialisierten Gewichten $\mathbf{w} = (w_0, \ldots, w_3)$. Das Training wird nach 60 Zeitschritten beendet. Dabei ist der quadratische Fehler auf unter 10^{-5} gesunken. In Abbildung 5.4 ist der Verlauf des Versuchs abgebildet. Das positive Ergebnis des Versuchs war insofern zu erwarten, als dass das lernende System sehr gut zu der Struktur des zu identifizierenden Systems passt. Betrachtet man den Verlauf der gelernten Gewichte $\mathbf{w}$, so stellt man fast eine exakte Übereinstimmung mit den Koeffizienten c_i fest, mit Ausnahme des vierten Gewichts. Es wird nicht benötigt und vom Gradientenverfahren nach einigen Zeitschritten auf Null geregelt.

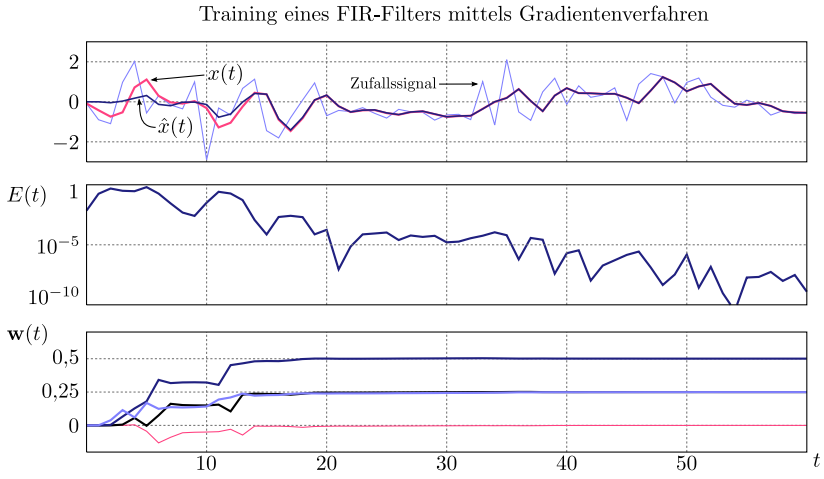

Abbildung 5.4: Training eines adaptiven Filters zur Systemidentifikation. Dem Filter wird ein zufälliges Signal präsentiert (hellblau) und die Filterausgabe (blau) mit der Ausgabe des zu identifizierenden Systems (rot) verglichen (siehe obere Grafik). Nach 20 Zeitschritten ist das Filter bereits sehr gut adaptiert.

5.1.5 Alternative Prädiktorarchitekturen

Viele vorherzusagende Prozesse haben Rückkopplungsschleifen und sind mithilfe adaptiver FIR-Filter nur ineffizient zu modellieren. Um eine hohe Präzision der Vorhersage rückgekoppelter Prozesse zu erreichen, braucht man bei der Verwendung von FIR-Filtern oft sehr viele Koeffizienten und somit auch mehr Rechenzeit- und Speicheraufwände. Daher kann es notwendig sein, die bisherige Prädiktorarchitektur zu erweitern.

Ein Filter mit lokaler Rekurrenz: Das Gamma-Filter

Die FIR-Synapse kann durch ein Gamma-Filter [36] ersetzt werden. Dieser spezielle Filtertyp hat ausschließlich lokale rekurrente Verbindungen und die angegebenen Lernregeln sind inhärent stabil. Das Filter besitzt nur einen zusätzlichen Parameter $\mu \in \mathbb{R}$ für die Regulation der Rückkopplungsstärke. Dabei ist die numerische Stabilität

des Filters garantiert, solange $0 < \mu < 2$. Für $\mu \in (0, 1)$ besitzt der verallgemeinerte *Verzögerungsoperator*

$$G(z) = \frac{\mu}{z - (1 - \mu)} \tag{5.9}$$

Tiefpasseigenschaften und für $\mu \in (1, 2)$ Hochpasseigenschaften. Wie in Abbildung 5.5 zu sehen, kann das Filter auch als eine Kaskade von einzelnen Filtern interpretiert werden. Beispielsweise würde für $\mu = 0{,}1$ das Ergebnis der Filterung eine gewichtete Summe aus verschiedenen tiefpassgefilterten Varianten desselben Signals sein. Dabei nimmt die Stärke der Filterung mit der Tiefe der Verzögerungskette zu. Die explizite zeitliche Einbettung wird damit aufgelöst. Für den Fall, dass der Parameter μ für jedes Modul separat gelernt wird, kann nicht wie bisher eine Verzögerungskette pro Sensorkanal für alle Prädiktormodule verwendet werden. Der Inhalt der Verzögerungskette ist dann spezifisch für jedes Modul. In kommenden Untersuchungen bliebe zu klären, wie eine Verzögerungskette mit statischen (d. h. nicht zu lernenden rekurrenten Verbindungen) die Vorhersageleistung beeinflusst. Im Gegensatz zu wenigen Anzapfungen und erlernbarer Rekurrenz stünden dann statische Rekurrenzen mit mehreren Anzapfungen, um ein gleiches Spektrum abzudecken. Damit wäre die Verzögerungskette wieder für alle Prädiktormodule gleichermaßen nutzbar.

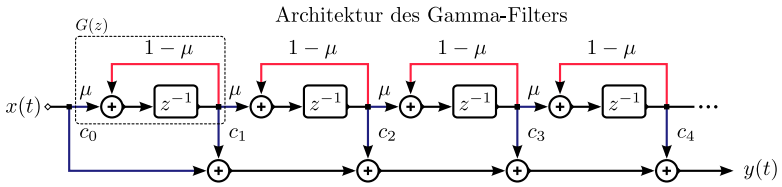

Abbildung 5.5: Schematischer Aufbau eines Gamma-Filters. Das Gamma-Filter besteht aus einer anzapfbaren Verzögerungsleitung mit lokalen rekurrenten Verbindungen. Der adaptive Parameter μ regelt die Art und Stärke der Rückkopplung. Werte die tiefer in die rückgekoppelte Verzögerungskette rutschen werden somit zunehmend stärker gefiltert.

Elman-Netzarchitektur

Ein ganz anderer Ansatz für einen Vorhersagemechanismus ist die Verwendung einer Elman-Architektur [10]. Ein solches Netz besteht aus Eingabeneuronen, einer verdeckten Neuronenschicht und den Ausgabeneuronen. Zusätzlich werden sogenannte *Kontextneuronen* verwendet, welche dazu dienen, den letzten Zustand der verdeckten Neuronen vorzuhalten. Demnach existieren genauso viele Kontextneuronen wie es verdeckte Neuronen gibt. Im Gegensatz zu den anderen Architekturen wird das komplette Elman-Netz in einem diskreten Zeitschritt durchgerechnet. Gelernt werden dabei nur die vorwärtsgerichteten synaptischen Verbindungen mithilfe der in Abschnitt 3.2.2 vorgestellten Fehlerrückführung. Die Verbindungen zu den Kontextneuronen werden dabei genauso wie die Synapsen der Eingabeschicht gelernt. Die Abbildung 5.6 zeigt ein solches Elman-Netz.

Das Kopieren der verdeckten Inhalte in die Kontextneuronen kann als statische rekurrente Verbindung interpretiert werden. Die Kontextneuronen werden bei dem in

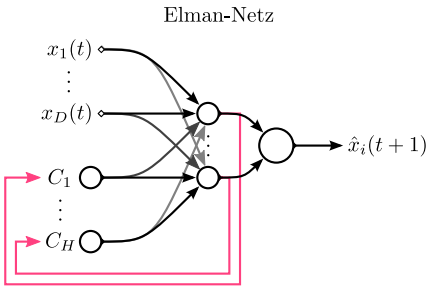

Abbildung 5.6: Schematischer Aufbau eines Prädiktormoduls mit Elman-Architektur.

Abschnitt 3.1.1 definierten Neuronenmodell mit Null initialisiert. Das Netz ist durch die rekurrenten Verbindungen mit einem Gedächtnis ausgestattet. Die Zeit ist hier also implizit repräsentiert.

Die Anzahl der verdeckten bzw. Kontextneurone muss dabei je nach gewünschter *Güte* skaliert werden, wobei zu beachten ist, dass der Rechenzeit- und Speicheraufwand dabei quadratisch mit der Anzahl der verdeckten Neuronen ansteigt. Sei N die Anzahl der Eingabe- bzw. Ausgabeneuronen und H die Anzahl der verdeckten und Kontextneuronen, so müssen $H^2 + 2NH$ Synapsen verarbeitet werden. Allerdings wird auch keine Verzögerungskette benötigt. Jedes Modul hat damit sein eigenes Gedächtnis.

5.1.6 Zusammenfassung und Fazit

Für die Vorhersage zukünftiger Sensorwerte aus den bisher gesammelten Informationen stehen eine Reihe an Verfahren zur Verfügung, von denen hier nur ein kleiner Anteil betrachtet werden konnte. Dabei wurden im Wesentlichen konnektionistische Architekturen untersucht. Die Voraussetzung dabei war für das Lernen ein Gradientenverfahren erster Ordnung zu verwenden und den Blickwinkel auf rechnerisch günstige und numerisch inhärent stabile Verfahren zu lenken. Dazu wird auf vollständig rekurrente Netzarchitekturen verzichtet und nur lokale bzw. statische Rekurrenz verwendet. Der sensorische Zustandsraum wird zeitlich expandiert, wobei dies wahlweise durch Verzögerungsketten oder rekurrente Verbindungen implementiert ist. Eine nichtlineare Expansion [20] der Sensoreingaben findet hierbei bisher nicht statt und bleibt daher für künftige Untersuchungen vorbehalten. Eine Untersuchung der Ansätze aus dem *Reservoir Computing* (vgl. Abschnitt 3.1.2) findet ebenfalls nicht statt, da trotz überzeugender Vorhersageleistungen eine Interpretation der gelernten Gewichte als Anzapfung eines nichtlinearen Dynamik-Reservoirs kaum möglich erscheint.

Ein umfassender Vergleich der vorgestellten Vorhersagemechanismen gestaltet sich mangels einheitlicher Testszenarien als schwierig und wird im Zuge dieser Arbeit nicht durchgeführt. Die Performanz des Prädiktors ist zudem maßgeblich von der Wahl seiner Parameter und der Ausgestaltung der zu prädizierenden Testdaten abhängig. Außerdem ist nicht zu garantieren, dass ein Gradientenverfahren die korrekte Einstellung der Gewichte findet [6], obwohl der Prädiktor aufgrund seiner Architektur prinzipiell

in der Lage wäre die Daten angemessen vorherzusagen, wenn die Gewichte ersteinmal korrekt eingestellt sind.

Bevorzugt wird das FIR-Prädiktormodul, da es leicht implementierbar, robust sowie sparsam im Verbrauch von Ressourcen ist. Die Qualität der Vorhersagen ist für den in dieser Arbeit anvisierten Verwendungszweck hinreichend gut. Im Hinblick auf die Multi-Experten-Architektur ist es sogar deutlich sparsamer, weil ein und dieselbe Verzögerungskette für alle Experten verwendet werden kann. Die erlernten Gewichte geben Aufschluss über Art und Weise bzw. die Intensität der Nutzung von Sensoreingaben. Somit ließen sich prinzipiell gezielt nutzlose Verbindungen ausfindig machen, welche entfernt werden können, um somit die Komplexität zu reduzieren, ohne das Ergebnis zu beeinträchtigen.

5.2 Ein wachsendes Experten-Gas

Das in Abschnitt 3.2.3 vorgestellte wachsende neuronale Gas (GNG) ist speziell für die Verwendung stochastischer Signale ausgelegt. Liegen nun beim *Online*-Lernen noch keine vollständigen Zeitreihen vor, welche dem Algorithmus in randomisierter Reihenfolge präsentiert werden könnten, so ergeben sich daraus einige Schwierigkeiten. Gegeben ist beispielsweise der zweidimensionale Eingaberaum aus den absoluten Winkelpositionen eines einfachen Roboterarms. Möchte man nun eine Anpassung des GNG in jedem Zeitschritt, aber ändern sich die Winkelpositionen des Arms vergleichsweise langsam oder für eine gewisse Zeit überhaupt nicht, müssen Vorkehrungen getroffen werden, um das GNG an diese besondere Situation anzupassen. Die Änderungen, die der Algorithmus an der Netzarchitektur vornimmt, sind in der bisherigen Variante bisweilen *zeitabhängig*. Das führt dazu, dass ein Netzumbau vorgenommen wird, wo eigentlich keiner erforderlich wäre und somit bereits Erlerntes schnell wieder *vergessen* wird. Würde man diesem Umstand mit einer verringerten Lernrate begegnen, um den Vergessensprozess zu verlangsamen, wird das System schnell träge und neuartige sensorische Information, welche nur sporadisch auftritt, kann nicht adäquat verarbeitet werden. Kurz: Das Dilemma zwischen Stabilität und Plastizität muss hier gelöst werden. Als erster Schritt muss die Zeitabhängigkeit der Prozesse abgemildert werden und durch Bedingungen ersetzt werden, welche unmittelbar am Lernprozess beteiligt sind.

Im Folgenden wird eine angepasste Version des GNG-U vorgestellt, um das Einfügen und Rationalisieren von verschiedenen Expertenmodulen zu regulieren. Dazu wurde der Originalalgorithmus modifiziert, in der Anzahl der verwendeten Parameter und Rechenkomplexität reduziert und besonders die zeitabhängigen Prozesse systematisch umfunktioniert, sodass sich die Abhängigkeit auf den Lernprozess selbst abbildet.

Das GNG-U bildet eine exzellente Ausgangsbasis für die Verwaltung der Expertenmodule. Betrachtet man jeden GNG-Knoten als einen Experten so bilden die Kanten ein Gerüst, welches für die Prädiktion oder die Auswahl der Experten bisher keine Funktion hat, aber wichtige Information über die Relevanz einzelner Expertenmodule enthält. Ein stark vernetzter Experte mit *jungen Kanten* ist somit von zentraler Bedeutung für ein System, wobei schwache Vernetzung und alte Kanten signalisieren, dass ein Experte an Funktion verloren hat und wegrationalisiert werden kann.

Für die angepasste Version wurde ein im GNG verdrängter Gedanke wiederbelebt. Das Abkühlen der Lernrate, ursprünglich im GNG abgeschafft, aber in anderen neuronalen Gasen noch enthalten, wird verwendet, um den Lernfortschritt zu signalisieren und eine Konvergenz zu erwirken. Die dynamische Lernrate $\epsilon(t)$ beginnt mit dem Startwert ϵ_0 und fällt dann ab, bis sie schließlich auf dem Plateau ϵ_R (einer minimalen Rest-Lernrate) liegen bleibt. Die Lernrate sinkt, um ein ständiges Umlernen auszubremsen, aber nicht gänzlich zu stoppen. Der wichtige Aspekt hierbei ist, dass jeder Experte seine eigene Lernrate hat und diese auch nur reduziert wird, wenn der Experte auch Gewinner war und wirklich seine Gewichte adaptiert hat. Die Intensität des Absinkens wird quantitativ an die Änderung der Gewichte gekoppelt. Eine große Änderung an den Gewichten führt zu einem raschen Absinken der Lernrate. Müssen die Gewichte nicht angepasst werden, weil alles in bester Ordnung ist, so geht auch die Lernrate nicht zurück. Somit entspricht die Lernrate einer Art Lernkontingent, welches sich zwar *aufbraucht* aber nie zur Gänze erschöpft ist. Interessant wäre in diesem Zusammenhang auch die Frage, welche Auswirkungen eine sich langsam wieder auffüllende Lernrate, also ein wiederaufladbares Lernkontingent hat. Die Aktualisierungsvoschrift für die dynamische Lernrate des gewinnenden Expertenmoduls s_1 ist

$$\epsilon_{s_1}(t) = \epsilon_{s_1}(t-1) \cdot e^{-\kappa \|\Delta W\|} \tag{5.10}$$

mit den Änderungen an der Gewichtsmatrix $\|\Delta W\|$. Die gesamte Lernrate, welche zur Adaption der Gewichte verwendet wird, ist dann

$$\eta_{s_1}(t) = \epsilon_{s_1}(t) + \epsilon_R \tag{5.11}$$

also die Summe aus der dynamischen Lernrate ϵ_{s_1} und der minimalen Rest-Lernrate ϵ_R. Der Parameter $\kappa \in \mathbb{R}$ regelt die Stärke der Abkühlung, wobei $\kappa > 0$ gelten muss. Er entspricht somit einer globalen Lernrate für das Expertengas. Je höher κ desto schneller können neue Experten eingefügt werden.

Die beschränkte Rationalität (vgl. Abschnitt 2.2) spielt bei der Anpassung des GNG eine besondere Rolle. Die maximale Anzahl der Experten ist als Parameter N_{max} vorgegeben und kann nicht überschritten werden. Begonnen wird wie üblich mit zwei Experten. Ursprünglich wurde das Einfügen neuer Knoten durch den Warteparameter λ reguliert, welcher nach einer bestimmten vergangenen Zeit das Einfügen neuer Knoten erlaubt. Diese Wartezeit wird abgelöst und durch eine mehr am Lernfortschritt orientierte Bedingung ersetzt. Erst wenn festgestellt wird, dass der aktuelle Gewinnerexperte ausgelernt hat, d. h. wenn die Bedingung

$$\epsilon_{s_1} < \epsilon_R \tag{5.12}$$

erfüllt ist, kann eine Änderung an der Netzarchitektur vorgenommen werden. Erst dann können neue Experten eingefügt werden, bestehende Kanten altern oder Experten entfernt werden. Der Parameter ϵ_R hat also eine weitere Funktion. Die Bedingung (5.12) blockiert die Änderungen der Struktur solange wie nötig ist, um ungestört zu lernen. Stößt ein Experte in eine neue sensomotorische Situation vor und ist somit ständig Gewinner, so braucht er sein Lernkontingent schneller auf und ein neuer Experte wird zeitnah hinzugezogen. Sind bereits viele Experten vertreten und das Lernen

durch häufige Experten-Wechsel verteilt, so verlangsamt sich der Einfügeprozess entsprechend.

Es folgen weitere Anpassungen des Algorithmus, diese sind in direktem Vergleich zum Original zu betrachten (vergleiche dazu Abschnitt 3.2.3). In Tabelle 5.1 sind die verwendeten Parameter und deren Standardwerte aufgelistet (vergleiche ebenfalls dazu die Parameter des original GNG-U in Tabelle 3.1 auf Seite 27).

- Nur der Gewinner s_1 adaptiert. Die Anpassung der Nachbarschaft von s_1 wird abgeschafft (hartes Wettbewerbslernen). Dies reduziert den Rechenaufwand um eine Nachbarschaftsbestimmung und der (vergleichsweise unkritische) Parameter ϵ_n, die Lernrate der Nachbarn, entfällt.

- Der Experte i mit der geringsten Nützlichkeit wird *nicht* sofort entfernt, sondern es werden lediglich die Kanten, welche von i ausgehen gealtert. Sind nun alle Kanten von i abgestorben wird er zwangsläufig entfernt. Damit ist der kritische Parameter k aus der Löschbedingung (3.24) abgeschafft, allerdings bleibt trotzdem die Möglichkeit über das Maximalalter der Kanten a_{max} indirekt in die Geschwindigkeit der Graphenausdünnung einzugreifen. Die Nützlichkeit

$$\Delta U_{s_1}(t) = E_{s_2}(t) - E_{s_1}(t) \tag{5.13}$$
$$U_{s_1}(t) = 0{,}1\,\Delta U_{s_1}(t) + 0{,}9\,U_{s_1}(t-1) \tag{5.14}$$

  wird wie gehabt nur jeweils für den Gewinner ermittelt und mit leichter Tiefpassfilterung aktualisiert.

- Die von s_1 ausgehenden Kanten werden nicht mehr gealtert. Es wird auch nicht in jedem Zeitschritt über alle Kanten gesucht, um diejenigen mit überschrittenem Verfallsdatum zu entfernen, sondern es wird in jedem Zeitschritt nur eine Kante stichprobenartig überprüft. Der Parameter a_{max} ist vergleichsweise unkritisch und erlaubt diesen Reduktionsschritt.

- Neue Experten werden nicht zwischen den beiden Knoten mit dem höchsten akkumulierten Fehler eingefügt, sondern dort, wo das Lernkontingent gerade aufgebraucht wurde und was somit zur *Anforderung von Verstärkung* geführt hat. Die Werte der Gewichte werden durch eine schnell adaptierende Erkundungseinheit [27] vorbereitet und vom neu eingefügten Experten übernommen bzw. unter Symmetriebrechung von dem geteilten Experten kopiert. Der akkumulierte Fehler und die Nützlichkeit werden vom aktuellen Gewinner übernommen, wobei der Fehler bei beiden halbiert wird. Somit entfällt auch α.

ϵ_0	ϵ_R	κ	a_{max}
0,1	0,01	0,5	88

Tabelle 5.1: Standard-Parameter des modifizierten GNG-U für die Verwendung zur Expertenverwaltung.

Trennung von statischen und dynamischen Anteilen

Die Experten (FIR-Prädiktoren) arbeiten, wie in Abschnitt 5.1.4 beschrieben, auf den zeitlich expandierten Sensordaten $\tilde{\mathbf{x}}(t) \in \mathbb{R}^{DK+1}$. Somit geht die aus Abbildung 3.5 auf Seite 29 bekannte grafische Darstellung verloren, da der Eingaberaum nun wesentlich größer ist. Bei Systemen mit $D \leq 3$ sensorischen Freiheitsgraden kann der noch nicht expandierte Eingabevektor $\mathbf{x}(t) \in \mathbb{R}^D$ bequem grafisch dargestellt werden. Solch eine grafische Darstellung ist durchaus erwünscht, vereinfacht sie doch die Interpretation der Daten enorm. So ist es wünschenswert, verschiedene Experten qualitativ wie bei einem neuronalen Gas den nicht expandierten Daten zuordnen zu können, nur dass diese, wenn mehrere dicht beieinander liegen, sich auch noch in den höheren Signaleigenschaften unterscheiden.

Eine Methode die Anschaulichkeit beizubehalten und die räumliche Spezialisierung der Experten in $\mathbb{R}^D$ zu erwirken ist, die Mittelwertschätzung (Abschnitt 5.1.3) und die FIR-Schätzung (Abschnitt 5.1.4) parallel auf denselben Daten durchzuführen und den Gesamtfehler neuzugestalten. Beide Submodule machen demnach ihre *eigene Schätzung*. Der Gesamtfehler ist dann das Produkt aus beiden Einzelfehlern. Das heißt, nur wenn beide Submodule einen genügend kleinen Fehler haben, ist der Zustand *wohl-detektiert*. Damit bleiben alle Vorteile erhalten: Erstens die präzise Schätzung des FIR-Prädiktors auf Basis der zeitlich expandieren Daten, welche räumlich durch den Mittelwertschätzer auf einen bestimmten Bereich im $\mathbb{R}^D$ spezialisiert wird. Und zweitens die Anschaulichkeit den Prädiktor durch einen quasistatischen Vektor in einem GNG-Graphen zu repräsentieren.

Experiment

In Abbildung 5.7 ist das anfängliche Wachstum des modifizierten GNG-U dargestellt. Es ist der Beginn des Experiments abgebildet, wobei unmittelbar nach jedem Einfügeprozess ein Zwischenstand festgehalten wurde. Anhand der blau gekennzeichneten Trajektorie kann man den Zustand des Sytems über die letzten 1000 Zeitschritte nachvollziehen, die Pfeilspitze stellt dazu den momentanen Zustand dar. Wie man erkennt, passiert das Einfügen neuer Experten unmittelbar dort, wo sich das System zur Zeit befindet und folglich Lernfortschritte erzielt.

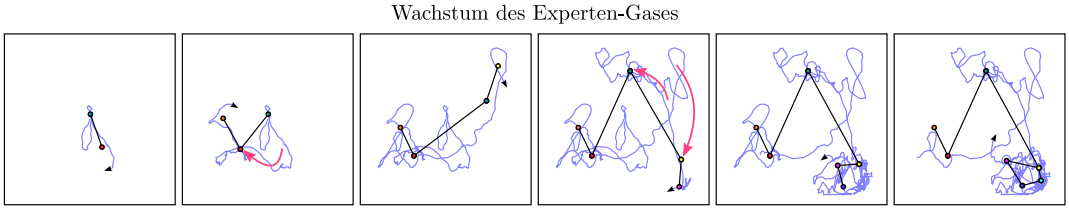

Abbildung 5.7: Dargestellt ist der Beginn eines wachsenden Experten-Gases. Das Einfügen neuer Experten passiert unmittelbar dort, wo sich das System zur Zeit befindet und aktuelle Lernfortschritte erzielt.

Zusammenfassung

Dieser Abschnitt stellte die Modifikationen des GNG-U vor, welche notwendig sind, um den Algorithmus an die Anforderungen realer Sensordaten anzupassen. Die zeitlichen Abhängigkeiten wurden dabei entfernt. Änderungen der Topologie vollzieht der Algorithmus nun in Abhängigkeit zur Änderung der Gewichte. Die Knoten (Experten) wurden als FIR-Prädiktormodule implementiert, damit auch dynamische Zustände unterschieden werden können. Jeder Experte hat sein eigenes dynamisches Lernkontingent. Insgesamt wurde die Anzahl der nötigen Parameter reduziert. Der wichtigste Parameter ist κ, welcher die Anpassungsgeschwindigkeit des gesamten neuronalen Gases bestimmt.

6 Entwurf eines Filters zur diskreten Differentiation

Dieser Abschnitt beschreibt den Aufbau und den vergleichenden Test von Differentiationsfiltern. Für die näherungsweise Ableitung einer diskreten Zeitreihe gibt es unterschiedliche Möglichkeiten der Realisierung von denen hier drei Verfahren exemplarisch untersucht und die jeweiligen Stärken, Schwächen und Einsatzmöglichkeiten genauer bestimmt werden sollen.

Die Ableitung einer diskreten Zeitreihe gestaltet sich mitunter schwierig, insbesondere wenn es sich dabei um reale Sensorwerte handelt. Diese haben im Allgemeinen einen für die Ableitung *nicht zu vernachlässigenden* störenden (meist hochfrequenten) Rauschanteil, welcher prinzipiell durch das Differenzieren noch verstärkt wird. Die Implementation eines verlässlichen und genauen diskreten Ableitungsfilters ist für diese Arbeit von besonderem Interesse, da im Kernstück der intrinsischen Motivation (siehe Abschnitt 2.4 und 4.4) der Vorhersagefehler der Prädiktion abgeleitet wird und als Lernsignal verwendet wird. Der Vorhersagefehler ist eine Funktion der Sensorwerte und daher schlägt sich der darin enthaltende Rauschanteil unmittelbar auf das Lernsignal nieder. Zeitgleich zur Ableitung muss also auch eine Tiefpassfilterung vorgenommen werden, um einer ungewollten Rauschverstärkung vorzubeugen.

Für die Anwendung als Lernsignal könnte eine genügend kleine Lernrate durch die innewohnende Tiefpasswirkung den Einfluss des Rauschens zwar kompensieren, allerdings schränkt man damit unnötig die Reaktivität des Lernens ein. Darüber hinaus bleibt die Schwierigkeit erhalten, das stark verrauschte Lernsignal richtig zu interpretieren. Bei der Verwendung adaptiver Lernraten besteht zudem die Gefahr, dass das eigentliche Lernverfahren dadurch instabil wird. Schließlich ist es auch wünschenswert in Abhängigkeit des im Signal vorhandenen Rauschens die Grenzfrequenz des Tiefpassanteils zu parameterisieren, um das Verfahren für die Anwendung mit unterschiedlichen Sensorqualitäten anzupassen.

6.1 Ableitung durch Differenzenquotienten

Gegeben seien die Werte $x(t) \in \mathbb{R}$ einer diskreten Zeitreihe zum Zeitpunkt t. Die einfachste Art diese näherungsweise abzuleiten ist die Bildung des Differenzenquotienten

$$y(t) = \frac{x(t) - x(t-1)}{\Delta t} \tag{6.1}$$

unter Verwendung des zeitlichen Abstands Δt der aufeinanderfolgenden *samples*, wobei $y(t)$ das Ergebnis der Ableitung ist. Dies ist äquivalent zu einer Hochpassfilterung 1. Ordnung. Bei Anwesenheit hochfrequenten Rauschens in der Zeitreihe sind die abgeleiteten Daten aber meist unbrauchbar. Die Abbildung 6.1 demonstriert die Aus-

wirkung von Gleichung (6.1) auf ein verrauschtes Testsignal. Als ein solches kommt eine Sinusschwingung $T(t) = \sin(t^2)$ mit stetig anwachsender Frequenz zum Einsatz. Dessen analytische Ableitung ist $T'(t) = 2t\cos(t^2)$. Dem Testsignal wird ein normalverteiltes Rauschen $\xi = N(0, 10^{-2})$ hinzugefügt, um die Auswirkungen der diskreten Differentiation zu demonstrieren. Wünschenswert wäre demnach ein Ergebnis, welches in diesem Fall möglichst dicht an der analytischen Lösung liegt.

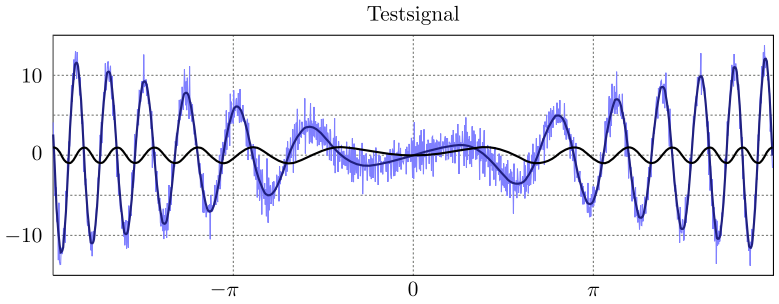

Abbildung 6.1: Abbildung eines leicht verrauschten Testsignals T (schwarz) dessen analytischer Ableitung T' (dunkelblau) und der stark verrauschten diskreten Ableitung (hellblau).

Das Hochpassfilter verstärkt per Definition höhere Frequenzen verhältnismäßig mehr als niedrigere. Somit kommt man zwangsläufig dazu, die Ableitung durch ein weiteres Glättungsfilter zu ergänzen, um eine ungewollte Verstärkung des Rauschanteils zu vermeiden. Bei linearen Filtern ist theoretisch die Reihenfolge der Filterung beliebig, d. h. es macht keinen Unterschied, ob man erst glättet und dann ableitet oder andersherum. Es können auch beide Eigenschaften in einem einzigen Filter vereint werden. Praktisch gesehen treten bei der Berechnung dennoch Unterschiede auf und man muss in jedem Fall die Art der Filterung an die Anforderungen der Anwendung anpassen. Für diese Arbeit wird ein kompaktes Kombinationsfilter bevorzugt.

6.2 Ableitung mittels Bandpassfilter

Die einfachste Lösung in die Ableitung eine Tiefpasseigenschaft zu integrieren ist das bestehende Filter um eine weitere Anzapfung zu erweitern, wodurch man ein einfaches nichtrekursives Bandpassfilter 2. Ordnung mit der Übertragungsfunktion

$$H_{BP}(z) = \frac{1 + z^{-2}}{2} \tag{6.2}$$

erhält, wobei $z = e^{i\omega}$ aus der Kreisfrequenz $\omega \in [0, \pi] \subset \mathbb{R}$ und der imaginären Einheit i besteht. Die Übertragungsfunktion des Bandpassfilters hat zwei Nullstellen bei $w_1 = 0$ und $w_2 = \pi$ (vgl. dazu Abbildung 6.2). Jedoch sind weder die Ableitung noch die Tiefpasswirkung annähernd ideal. Hervorzuheben ist allerdings der geringe Rechenaufwand und die fast vernachlässigbar kleine Signalverzögerung von nur einem Zeitschritt. Ein ideales Ableitungsfilter mit kombinierter Tiefpasswirkung hätte aber

eine wunschgemäße Übertragungsfunktion

$$H_{ideal}(e^{i\omega}) = \begin{cases} i\omega & |\omega| \leq \omega_c \\ 0 & \omega_c < |\omega| < \pi \end{cases} \tag{6.3}$$

mit einstellbarer Grenzfrequenz $f_c = \omega_c/2\pi$. Abbildung 6.2 zeigt die Frequenzgänge eines idealen *Tiefpass-Differentiators* und die ersten beiden Näherungen.

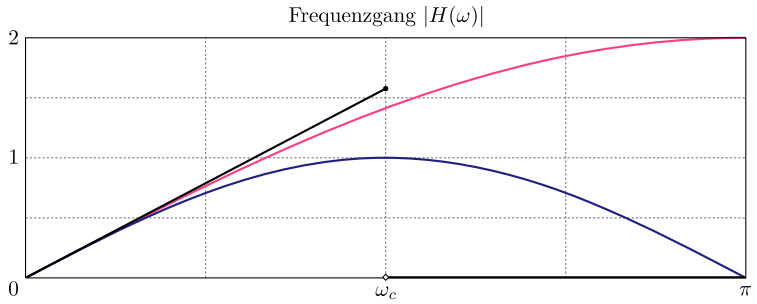

Abbildung 6.2: Die Abbildung zeigt die Frequenzgänge eines idealen Tiefpass-Differentiators (schwarz), eines Bandpassfilters 2. Ordnung (blau) und der einfachen Ableitung mittels Differenzenquotienten (rot).

6.3 Herleitung eines Tiefpass-Differentiators

Ein üblicher Ansatz ein Filter zu konstruieren, welches dem Gewünschten wesentlich näher kommt ist in [16] beschrieben. Dazu wird die ideale Übertragungsfunktion (6.3) durch eine Fourierreihe expandiert und danach die unendliche Anzahl an Koeffizienten auf eine endliche Anzahl N abgeschnitten. Das nichtrekursive Filter ist dann durch seine reellwertigen Koeffizienten

$$c_k = -\frac{1}{\pi}\left(\frac{\sin(k\omega_c)}{k^2} - \frac{\omega_c \cos(k\omega_c)}{k}\right) \tag{6.4}$$

gegeben, wobei der Index $k \in \mathbb{Z}$, $k = -K \ldots K$ und somit $N = 2K + 1$ die Anzahl der Koeffizienten ist. Das Filter besitzt damit eine vergleichsweise übersichtliche geschlossene Darstellung seiner Koeffizienten unter Angabe der Grenzkreisfrequenz ω_c. Man berechnet die Filterung dann mittels der üblichen Filtergleichung

$$y(t) = \sum_{k=-K}^{K} c_k x(t-k). \tag{6.5}$$

Das Abschneiden der unendlichen Reihe auf N Koeffizienten führt zu einer erhöhten Welligkeit der Übertragungsfunktion. Zur Abmilderung des sogenannten Gibbsschen Phänomens kann wahlweise die Anwendung der *Sigmafaktoren* oder einer *Fensterfunktion* (bspw. des Hamming-Fensters) durchgeführt werden. Dabei werden die abgeschnittenen Koeffizienten c_k mit ihren korrespondierenden Sigmafaktoren

$$\sigma_k = \frac{\sin(\pi k/K)}{\pi k/K} \tag{6.6}$$

multipliziert oder alternativ durch das Hamming-Fenster

$$w_k = 0{,}54 + 0{,}46\cos(\pi k/K) \tag{6.7}$$

betrachtet. In Abbildung 6.3 ist die Impulsantwort nach einer Gewichtung mit dem Hamming-Fenster für $N = 101$ und $\omega_c = \frac{\pi}{10}$ abgebildet. Abbildung 6.4 zeigt die resultierende Übertragungsfunktion und im Detail die Auswirkung der neu gewichteten Koeffizienten.

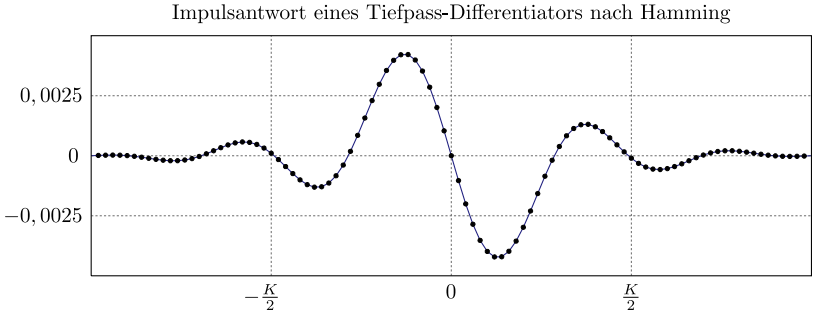

Abbildung 6.3: Impulsantwort des Hamming-Differentiators mit $N = 101$ Koeffizienten und Grenzkreisfrequenz $\omega_c = \frac{\pi}{10}$.

Wann die Reihe abgeschnitten wird ist stark von der Anwendung abhängig. Endlich muss die Reihe zwangsläufig sein, möchte man dieses Filter auch implementieren, doch je mehr Koeffizienten man zulässt, desto näher kommt das Ergebnis auch an die ideale Übertragungsfunktion heran. Möchte man bereits aufgenommene Daten filtern und hat dafür prinzipiell beliebig viel Zeit, so kann man bis zum gewünschten Grad der Genauigkeit die Anzahl der Koeffizienten erhöhen. In der Regel ist man hierbei mit $N = 101$ gut gerüstet. Für die Anwendung in Echtzeit ist die Filterlänge aber entscheidend. Das angegebene nichtrekursive Filter hat einen *linearen Phasengang* und somit eine konstante *Gruppenlaufzeit* von $\frac{1}{2}(N-1)$ Zeitschritten. So hat man bei einem *kausalen* Filter, d. h. ein Filter, das nur auf aktuellen und vergangenen Daten arbeitet, bei 100 Hz Abtastrate und $N = 101$ Koeffizienten eine effektive Signalverzögerung von $\frac{1}{2}(N-1) \cdot 10\text{ ms} = 500\text{ ms}$. Selbst wenn die Signalverzögerung für die Anwendung nicht von Bedeutung ist, so ist es möglicherweise die benötigte Rechenzeit, welche es erforderlich macht die Anzahl der Koeffizienten zu verringern. Im Allgemeinen werden für die Implementierung eines nichtrekursiven Filters jeweils N Multiplikationen und N Additionen pro Zeitschritt benötigt. Um weitere Rechenzeit einzusparen kann man bei Filtern mit symmetrischen (bzw. antisymmetrischen) Koeffizienten auf die Hälfte der Multiplikationen verzichten, indem man sich die Distributivität zunutze macht und bei der Implementierung der Faltung die zusammengehörigen verzögerten Samples $x(t)$ und $x(N-t)$ zuerst addiert (bzw. subtrahiert) und danach erst mit den Koeffizienten multipliziert.

Das in diesem Abschnitt gewonnene Filter überzeugt durch seine schlanke Darstellung und verständliche Herleitung [16]. Der einzig erwähnenswerte Nachteil dieses Filters ist, dass mit zunehmender Stopbandbreite, die Anzahl der erforderlichen Koeffizienten für viele Anwendungen untragbar hoch wird. Für die Anwendung in dieser

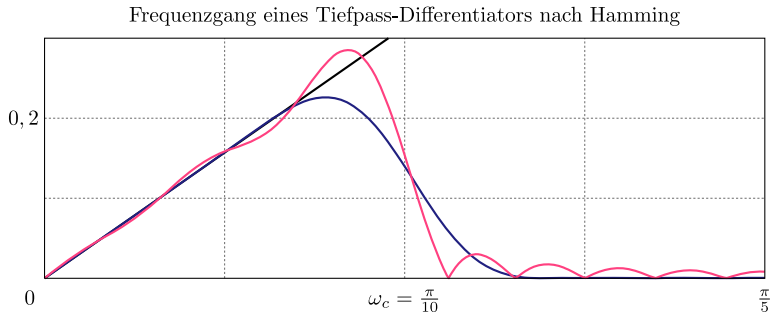

Abbildung 6.4: Frequenzgang des Hamming-Differentiators mit $N = 101$ Koeffizienten und Grenzkreisfrequenz $\omega_c = \frac{\pi}{10}$ (rot) und nach Anwendung des Hamming-Fensters (blau).

Arbeit ist es erforderlich, dass die Signalverzögerung 100 ms nicht wesentlich überschreitet. Damit wäre ein nichtrekursives Filter auf $N = 21$ Anzapfungen begrenzt. Weiterhin müssen für den Lernalgorithmus zu verschiedenen Signalen die Ableitungen gebildet werden, womit es auch Rechenzeit einzusparen gilt. Daher wird noch weiter untersucht, welches Filter bei näherungsweise gleicher Qualität der Ableitung und Tiefpassfilterung etwas sparsamer in der Anzahl der Filter-Koeffizienten ist. Zur namentlichen Unterscheidung werden die Filter nach ihren Erbauern benannt, sodass das bisher untersuchte als *Hamming*-Differentiator und das folgende als *Selesnick*-Differentiator betitelt wird.

Ein alternatives Ableitungsfilter

In [40] wird ein Ableitungsfilter beschrieben, welches im Wesentlichen vergleichbare Eigenschaften aufweist, wie das Filter aus dem vorangegangenen Abschnitt. Bei diesem Filter kann die Grenzfrequenz indirekt durch die Anzahl der Koeffizienten und die Wahl eines diskreten Parameters angegeben werden. Die Koeffizienten für den Selesnick-Differentiator berechnen sich durch

$$s_n = -\frac{2\pi}{N^2} \sin\left(\frac{2\pi}{N}\left(n - \frac{N-1}{2}\right)\right), \tag{6.8}$$

wobei die formale Darstellung aus der Literatur weitestgehend vereinfacht wurde, sodass sich für $N = 21$ die Grenzfrequenz $\omega_c \approx \frac{\pi}{10}$ ergibt.

6.4 Nachoptimierung und Analyse der Eigenschaften

Die beiden möglichen Kandidaten wurden in der Anzahl der Anzapfungen reduziert, sodass die maximal vertretbare Verzögerungszeit von 10 Zeitschritten eingehalten wird. Dabei stellt sich heraus, dass die Anwendung der Fensterfunktion (6.7) zur Glättung der Koeffizienten c_k beim Hamming-Differentiator das Wunschergebnis zu stark verfälscht und die Übertragungsfunktion dadurch wesentlich von der idealen Kennlinie abweicht. Durch einen manuell ermittelten Korrekturfaktor von $b = 2{,}091$ konnte das

Filter glücklicherweise aber nachjustiert werden, sodass sich wieder vergleichbare Eigenschaften ergeben. Als guter Kompromiss aus vergleichsweise wenigen Koeffizienten, einem großem und flachem Stopband und präzisen Ableitungseigenschaften wurden die Filter-Koeffizienten

$$q_k = b\, w_k c_k \tag{6.9}$$

ermittelt und in Abbildung 6.5 mit den bisherigen Varianten verglichen. Dabei stellt sich heraus, dass der nachoptimierte Hamming-Differentiator eine etwas geringere Flankensteilheit im Übergangsbereich erworben hat und einen nahezu konstantes Stopband mit starker Dämpfung bei circa 10^{-3}. Der Differentiator von Selesnick hat im Gegenzug dazu einen steileren Übergangsbereich und eine für höhere Frequenzen zunehmende Dämpfung. Der interessante Aspekt ist aber die quadratische Abweichung beider Kandidaten von der idealen Differentiatorkennlinie $H(\omega) = i\omega$. Nach der Optimierung zeigte der Hamming-Differentiator die geringsten Abweichungen.

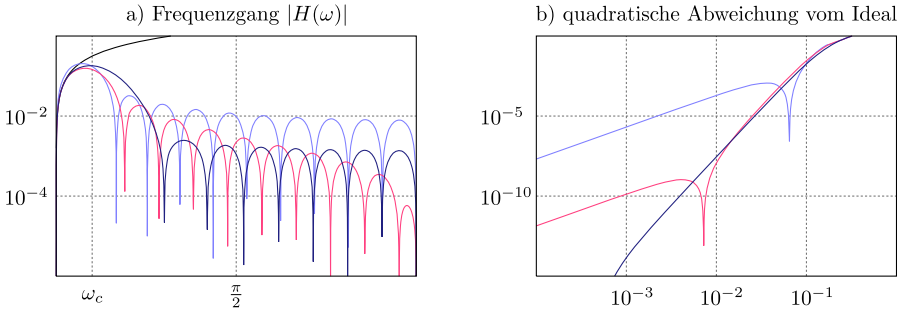

Abbildung 6.5: Vergleich von Frequenzgang und der quadratischen Abweichung zur idealen Kennlinie beim Hamming-Differentiator (hellblau), einer optimierten Fassung desselben (blau) und der Selesnick-Variante (rot).

Fazit

Für die Anwendung in dieser Arbeit wird die Hamming-Variante verwendet, da Wert auf die maximale Genauigkeit der Differentiation gelegt wird, welche noch unter den gegebenen zeitlichen Randbedingungen realisierbar ist. Die Dämpfung im Stopband ist bei allen Varianten ausreichend hoch. Für den Anwendungsfall sehr stark verrauschter Sensordaten fiele die Wahl auf die Selesnick-Variante. Die errechneten Koeffizienten für beide Varianten sind in Tabelle A.1 in Anhang A.2 gelistet.

7 Bewertung, Auswahl und Ausführung motorischer Aktionen

Der Zustand des Systems wird mithilfe der Experten identifiziert. Nun muss geklärt werden, wie motorische Aktionen zustande kommen. Dieses Kapitel beschreibt, wie Zustände und Aktionen bewertet werden, auf Basis dieser Bewertung neue Aktionen ausgesucht und diese schließlich ausgeführt werden. Der erste Abschnitt gibt eine kurze Einführung in die Thematik des bestärkenden Lernens. Dabei wird speziell die Funktionsweise einer Lernregel erläutert, welche für die Bewertung von Zuständen und Aktionen verwendet wird. Im zweiten Abschnitt werden Verfahren für die Auswahl der Aktionen untersucht und gezeigt, wie mit Hilfe einer homöostatischen Lernregeln das Verhalten des Individuums in einfacher Weise reguliert werden kann. Der letzte Abschnitt erläutert, wie aus der Auswahl diskreter Aktionen wieder kontinuierliche Motorwerte produziert werden.

7.1 Bestärkendes Lernen

Wurden im bisherigen Verlauf der Arbeit Lernverfahren beschrieben, welche entweder ein explizites Lernziel durch Vorgabe von Beispielen verfolgten oder ganz und gar sich selbst überlassen wurden (selbstorganisierend), so soll nun ein Vertreter einer neuen Klasse an Verfahren vorgestellt werden, welche auf interessante Weise zwischen diesen Welten wandelt. Bestärkendes Lernen (engl. *reinforcement learning*) beschreibt eine ganze Familie von Lernverfahren, welche in gewisser Weise biologisch inspiriert sind. Bestärkendes Lernen ist Lernen durch *Lohn und Strafe*. Was einmal Lehrer war, wird auf einen *Kritiker* reduziert. Dieser bewertet lediglich das Verhalten des Systems, kann aber keine expliziten Beispiele vorgeben. Gutes Verhalten soll belohnt und schlechtes bestraft werden. Dabei fällt die Unterteilung in Belohnung und Bestrafung meist unter den Tisch und wird durch eine skalare Funktion $r(t)$ beschrieben, welche beispielsweise einen hohen Wert annimmt, wenn die letzte Aktion *gut* war und einen niedrigeren Wert annimmt, wenn sie *schlecht* war. Dabei ist es wichtig, dass diese Funktion die Skala von schlecht bis gut möglichst gleichmäßig und monoton steigend abbildet. Wie bei der künstlichen Evolution steht und fällt auch hier der Erfolg des Verfahrens mit der Wahl der Fitnessfunktion, welche für das System als abstrakte numerische Belohnung gedeutet wird und allgemein als *Reward*-Funktion bezeichnet wird.

Das Lernziel des Individuums ist auf die Anhäufung von möglichst viel Belohnung definiert. Der Blick ist dabei in die Zukunft gerichtet ist und es gilt den Erwartungswert aller zukünftigen akkumulierten Belohnung zu maximieren. Dazu muss das Individuum die ihm zur Verfügung stehenden M Aktionen in den N verschiedenen Zuständen ausprobieren, die Bewertung einholen und auf dieser Grundlage eine Strategie entwickeln und verbessern. Folgt das System seiner bisherigen Strategie und bedient sich

der Aktion, welche aus dem aktuellen Zustand die höchste zu erwartende Belohnung für alle Folgezustände verspricht, so spricht man von Ausnutzung. Im Allgemeinen ist während des Lernens, aber vor allem zu Beginn, die Strategie zweifellos schlecht – wenn nicht sogar völlig falsch. Daher ist es essentiell, dass auch Aktionen ausprobiert (exploriert) werden, die auf Grundlage der aktuellen Strategie kein Lob versprechen. Häufige Exploration ist außerdem erforderlich, wenn die Vergabe von Belohnungssignalen höchst nicht-stationär ist oder sich zeitlich begrenzte Belohnungs-Nischen auftun. Dieser Kompromiss zwischen Exploration und Ausnutzung ist wahrlich ein Dilemma. Holt man sich eine vermutlich suboptimale Belohnung ab und ist damit zufrieden oder versucht man einen Glücksgriff, welcher eventuell mehr einbringt – aber auch viel weniger Belohnung ergeben könnte.

Entgegen der in der Literatur üblichen Art und Weise wird direkt die Erläuterung der verwendeten Lernregel vorangestellt und nur die für das Verständnis wichtigen Erklärungen folgen darauf. Auf eine umfassende Einführung in die Thematik bestärkendes Lernen wird aufgrund der Diversität an Literatur verzichtet und auf die entsprechenden Quellen verwiesen [45, 7, 38, 15].

Die Frage ist nun, wie das Individuum eine optimale Strategie finden kann, welche es ihm ermöglicht aus jedem Zustand heraus so zu handeln, dass daraus eine möglichst hohe Belohnung erwächst. Grundlage dieser Strategie ist die kontinuierliche Bewertung der besuchten Zustände und der dort ausgeübten Aktionen. Dafür wird für jeden Zustand $s \in \mathcal{S}$ und jede Aktion $a \in \mathcal{A}$ eine skalare Größe vorgehalten, welche aussagt, wie gut oder schlecht die jeweilige Aktion in dem jeweiligen Zustand ist. Diesen Wert nennt man *Aktionswert*. Demnach ergibt sich eine Matrix $\mathbf{Q} \in \mathbb{R}^{N \times M}$ in welcher alle Aktionswerte zusammengefasst werden.

SARSA-Lernregel

Die Q-Matrix bildet die Grundlage für eine Entscheidung. An den Aktionswerten kann abgelesen werden, welche Aktionen bisher für gut oder schlecht befunden wurden. Den Aufbau und die Aktualisierung dieser Matrix leistet die SARSA-Lernregel

$$\Delta Q(s(t), a(t)) = \alpha\Big(r(t) + \gamma Q\big(s(t+1), a(t+1)\big) - Q\big(s(t), a(t)\big)\Big) \tag{7.1}$$

mit der Lernrate $\alpha \in \mathbb{R}$ und dem sogenannten Diskontierungsfaktor $\gamma \in \mathbb{R}$. Die Lernrate wird wie üblich $0 < \alpha \ll 1$ gewählt, wohingegen der Diskontierungsfaktor normalerweise $0 \ll \gamma < 1$, d. h. nahe 1 gewählt wird. Der Zustand $s(t+1)$ ist der Folgezustand von $s(t)$, welcher durch ausführen von $a(t)$ erreicht wurde und $a(t+1)$ ist die Aktion, welche wiederum in $s(t+1)$ ausgewählt wurde. Die Funktionsweise der Lernregel kann unterschiedlich interpretiert werden. Die allgemein übliche Interpretation ist es die Anpassung als Fehlerminimierung zu betrachten, also den Fehler $E_Q = Q_{soll} - Q_{ist}$ zu minimieren. Dabei ist $Q_{ist} = Q\big(s(t), a(t)\big)$ der aktuelle Aktionswert und $Q_{soll} = r(t) + \gamma Q\big(s(t+1), a(t+1)\big)$ die erhaltene Belohnung plus einer Schätzung über die zukünftig zu erwartende Belohnung. Dabei regelt γ wie stark die Schätzung über zukünftige Werte mit einbezogen werden soll. Die Lernrate α regelt wie üblich die Stärke der Adaption. Anders interpretiert kann man $\alpha\left(r(t) + \gamma Q\big(s(t+1), a(t+1)\big)\right)$ als Eingabeterm betrachten, welcher die Belohnungs-

information akkumuliert, wohingegen $-\alpha Q(s(t), a(t))$ als Verfallsterm betrachtet werden kann, welcher den Aktionswert exponentiell absinken lässt, falls keine weiteren Belohnungen mehr eintreffen.

Das Quintupel $(s(t), a(t), r(t), s(t+1), a(t+1))$ zur Ausführung der Lernregel ist Namensgeber des Verfahrens. Die SARSA-Lernregel gehört zur Klasse der sogenannten *On-Policy*-Algorithmen, d. h. das Individuum verbessert diejenige Strategie, die es auch gerade für die Auswahl der Aktionen verwendet. Für Anwendungen dehnen ein explitzites Training vorausgeht (also kein *Online*-Lernen), ist es demnach auch denkbar, eine Strategie zu suchen und zu verbessern, welche für die Zeit des Trainings nicht verwendet wird. Da es aber beim Lernen ohne zeitliche Einschränkung keine explizite Trainingsphase gibt, wird nur diese eine Strategie vewendet (vgl. dazu Abschnitt 3.2.2).

Auf der Spur der Förderungswürdigen

In der bisherigen Form bewertet das Lernverfahren rückwirkend das letzte Paar aus Zustand und Aktion nach der erhaltenen Belohnung und somit nach der Güte der Aktionsauswahl. Die vorangegangenen Zustände finden dabei keine Berücksichtigung, auch wenn sie indirekt zum Erreichen der aktuellen Belohnung oder Bestrafung beigetragen haben. Um nun auch diese Zustände am Erfolg bzw. Misserfolg teilhaben zu lassen, verwendet man einen einfachen Mechanismus, die sogenannten *eligibility traces*. Dazu werden die kürzlich besuchten Zustände und ihre dort ausgewählten Aktionen *markiert* und zur Aufteilung der Belohnung hinzugezogen. Mit Hilfe einer weiteren Matrix $\mathbf{e} \in \mathbb{R}^{N \times M}$ und der Aktualisierungsvorschrift

$$e(s,a) = \begin{cases} 1 & \text{wenn } s = s(t) \\ \lambda\gamma e(s,a) & \text{sonst} \end{cases} \quad \forall s,\, a \tag{7.2}$$

merkt man sich die besuchten Zustände. Die Lernregel wird nun dahingehend angepasst, dass in jedem Schritt die vollständige Q-Matrix aktualisiert wird.

$$\delta(t) = r(t) + \gamma Q(s(t+1), a(t+1)) - Q(s(t), a(t)) \tag{7.3}$$

$$\Delta Q(s,a) = \alpha\delta(t)\, e(s,a) \quad \forall s,\, a \tag{7.4}$$

Der Parameter λ lässt dabei die erzeugte Spur wieder verwischen, wenn sie altert. Die Anwendung der *eligibility traces* wird konventionell durch SARSA(λ) gekennzeichnet. Es folgt eine Übersicht über den vollständigen Algorithmus.

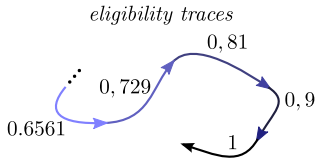

Abbildung 7.1: Illustration der Auswirkung von *eligibilty traces*.

SARSA(λ)-Algorithmus

$Q(s,a) \leftarrow$ zufällig $\forall s, a$
$e(s,a) \leftarrow 0\ \forall s, a$
loop
  Aktion $a(t)$ ausüben
  Zustand $s(t+1)$ feststellen
  Belohnung $r(t) = r(s(t+1))$ erhalten
  neue Aktion $a(t+1)$ auswählen
  $\delta(t) \leftarrow r(t) + \gamma Q(s(t+1), a(t+1)) - Q(s(t), a(t))$
  $e(s(t), a(t)) \leftarrow 1$
  for all s, a **do**
    $Q(s,a) \leftarrow Q(s,a) + \alpha\delta(t)e(s,a)$
    $e(s,a) \leftarrow \gamma\lambda e(s,a)$
  end for
  $s(t) \leftarrow s(t+1)$
  $a(t) \leftarrow a(t+1)$
end loop

7.2 Verfahren für die Aktionsauswahl

Gegeben ist also eine diskrete Anzahl von Aktionen und die zugehörigen Aktionswerte (Q-Matrix), welche zur aktuellen sensorischen Situation, d. h. zum aktuellen erkannten Zustand, die Bewertung für jede der zur Auswahl stehenden Aktionen enthalten. Nun ist es eine Frage der Strategie, wie aus den Aktionswerten eine Aktionsauswahl getroffen wird. Eine besonders gierige Strategie wäre es, immer die Aktion auszuführen, welche den maximalen Aktionswert hat (*Winner-Takes-All*-Regel), d. h. die Bedingung

$$a(t) = \arg\max_a Q(s(t), a) \tag{7.5}$$

erfüllt und somit zur vermeintlich höchsten Belohnung führt. Allerdings ist besonders am Anfang des Lernens die Abschätzung über die zu erwartende Belohnung noch ziemlich schlecht, was schnell dazu führt, dass möglicherweise bessere Aktionen vernachlässigt werden. Eine zu schnelle Festlegung auf eine bestimmte *beste* Aktion endet folglich in einem nicht-optimalen Verhalten.

Epsilon-Greedy-Strategie

Um die Gefahr nicht-optimaler Strategien zu reduzieren, muss die Auftretenswahrscheinlichkeit für andere Aktionen erhöht werden. Dadurch wird exploriert und die Möglichkeit für andere Aktionen gegeben, sich in der aktuellen Situation zu beweisen. Eine beliebte Art dies zu arrangieren ist die ε-*Greedy*-Strategie [45]. Dazu legt man eine kleine Konstante $\varepsilon \in \mathbb{R}$ mit $0 < \varepsilon \ll 1$ fest, die der Wahrscheinlichkeit entspricht, eine andere, d. h. nicht die scheinbar beste Aktion auszuwählen. Diese wird dann rein zufällig, d. h. ohne weitere Betrachtung der Aktionswerte, aus dem Spektrum möglicher Aktionen gelost.

7.2.1 Boltzmann-Selektion

Eine weitere typische Art eine Aktion a_i ($i = 0 \ldots M-1$) auszuwählen ist die sogenannte *Boltzmann-Selektion* [45, 7]. Dabei werden Auswahlwahrscheinlichkeiten $P(a_i|s)$ anhand ihrer Aktionswerte $Q(s, a_i)$ bestimmt, und zwar so, dass höhere Aktionswerte höhere Auswahlwahrscheinlichkeiten haben. Die Aktionswerte sind reell und für die Auswahlwahrscheinlichkeiten muss

$$\sum_{i=0}^{M-1} P(a_i|s) = 1 \tag{7.6}$$

mit $0 \leq P(a_i|s) \leq 1$ gelten, d. h. die Summe aller Einzel-Wahrscheinlichkeiten der jeweiligen Aktionen muss gleich Eins sein. Daher wird der Vektor aus beliebig reellen Aktionswerten mithilfe der *Softmax*-Funktion so umgeformt, dass sie als die Auswahlwahrscheinlichkeiten der Aktionswerte interpretiert werden können. Die Wahrscheinlichkeit im Zustand s die Aktion a_i auszuwählen ist dann durch

$$P(a_i|s) = \frac{e^{\beta Q(s,a_i)}}{\sum_{j=0}^{M-1} e^{\beta Q(s,a_j)}} \tag{7.7}$$

gegeben, wobei $\beta \in \mathbb{R}$ und $0 \leq \beta < \infty$. Der vorerst konstante Parameter β wird als *inverse Temperatur* bezeichnet und regelt die Zufälligkeit der Aktionsauswahl. Lässt man β gegen 0 gehen, so nähert sich die Verteilung einer Gleichverteilung und jede Aktion hat die gleiche Wahrscheinlichkeit $P(a_i|s) = \frac{1}{M}$. Die Aktionsauswahl wird zur rein zufälligen Auslosung. Für größer werdendes β verfestigt sich die Aktionsauswahl und verliert an Zufälligkeit. Im Grenzfall $\beta \to \infty$ entspricht es der gierigen Auswahl angegeben durch Gleichung 7.5.

Überlaufprävention der Exponentialfunktion

Der Wertebereich der Aktionswerte ist nicht beschränkt und da die Exponentialfunktion für positive Argumente sehr schnell gegen ∞ strebt müssen bei einer Implementation der Boltzmann-Selektion Vorkehrungen getroffen werden, um einen Variablenüberlauf zu vermeiden. Dazu verwandelt man den Überlauf in einen Unterlauf, welcher bei Gleitkommazahlen ein Streben gegen kleine Werte vorzeitig zu Null rundet. Addiert man vor der Anwendung der *Softmax*-Funktion einen beliebigen aber konstanten Wert c zu den *temperierten* Aktionswerten $q_i = \beta Q(s, a_i)$, so bemerkt man nach einigen einfachen arithmetischen Umformungen, dass sich am Ergebnis, d. h. an der Wahrscheinlichkeitsverteilung

$$p_i = \frac{e^{(q_i+c)}}{\sum_j e^{(q_j+c)}} = \frac{e^{q_i} e^c}{\sum_j e^{q_j} e^c} = \frac{e^c}{e^c} \frac{e^{q_i}}{\sum_j e^{q_j}} = \frac{e^{q_i}}{\sum_j e^{q_j}}$$

nichts geändert hat. Diese Eigenschaft der Exponentialfunktion kann daher verwendet werden, um gezielt den Wertebereich so zu verschieben, dass die Zahlen nicht groß sondern eher klein werden. Dazu ermittelt man $q_{max} = \max_i(q_i)$ und führt die *Softmax*-Funktion auf den präparierten Werten $\tilde{q}_i = q_i - q_{max}$ aus. Somit sind evtl. auftretende *zu große* Werte q_i entschärft.

Selbstregulation der inversen Temperatur

Für die Anwendung der Boltzmann-Selektion bleibt zu klären, in welcher Weise der Parameter β eingestellt wird. Eine konstante Voreinstellung zu finden stellt sich als schwierig heraus, da die Höhe der Aktionswerte stark von der *Reward*-Funktion und dem Lernverlauf abhängen. Weiterhin ist das optimale Intervall für eine konstante Einstellung durchaus sehr schmal und somit die Gefahr groß den Lernerfolg zu bremsen, weil entweder die Auswahl der Aktionen komplett gleichverteilt und somit zufällig ist oder ständig nur die vermeintlich beste Aktion ausgeführt wird und keine Exploration stattfindet. Die Anwendung der Boltzmann-Selektion ergibt also insofern nur einen Sinn, wenn der Parameter β adaptiv geregelt wird.

Im Rahmen dieser Arbeit hat sich herausgestellt, dass die Varianz der Wahrscheinlichkeitsverteilung selbst einen Lösungsansatz für die Regelung von β bereithält. Ist die Varianz zu klein, d. h. die sind Aktionswerte zu gleichverteilt, so kann man β ein wenig erhöhen und die Gleichverteilung löst sich auf. Ist die Varianz zu hoch, ist die Aktionsauswahl nicht ausgewogen genug und β sollte verringert werden. Mögliche Werte für die Varianz $v = \mathrm{Var}(P)$ sind auf das Intervall $[0, \frac{1}{M}]$ begrenzt. Daher liegt es nahe die Varianz vorerst auf den Wert $\frac{1}{2M}$ einzuregeln. Hierfür wurde eine einfache Lernregel

$$\beta_s(t) = \beta_s(t-1) \cdot \left(\frac{3}{2} - vM\right) \tag{7.8}$$

erstellt, welche genau dies leistet. Das Produkt vM kann nur Werte aus $[0, 1]$ annehmen. Womit β zur Absenkung minimal halbiert und zur Anhebung maximal um die Hälfte erhöht werden kann. Die Regelung erfolgt stabil innerhalb weniger Zeitschritte. Diese Lernregel hat unverkennbar einen homöostatischen Charakter (vgl. dazu Abschnitt 2.3). Sie hält die Verteilung der zur Auswahl stehenden Aktionen in einem ausgewogenen Verhältnis. Weiterhin ist die Berechnung vergleichsweise günstig. Die Varianz v der Aktionswahrscheinlichkeiten errechnet sich einfach durch

$$v(t) = \sum_i \left(P(a_i|s(t)) - 1/M\right)^2 \tag{7.9}$$

Anmerkung: Die Regelung über die Varianz hat bedauerlicherweise einen kleine Makel. Für den äußerst seltenen Fall, dass zwei oder mehr der reellwertigen (temperierten) Aktionswert exakt denselben Zahlenwert haben und diese auch gleichzeitig die höchsten Aktionswerte aus der Liste sind, wird die Lernregel instabil. Beispielsweise seien zwei der Aktionswerte $q_1 = q_2 = q_{max}$. Nach Anwendung der Lernregel haben die entsprechenden Wahrscheinlichkeiten nach einigen Zeitschritten wie zu erwarten ist die Werte $P_1 = P_2 = 0{,}5$ erreicht, wobei der Rest zu Null wird. Die resultierende Varianz kann dabei aber den Sollwert $\frac{1}{2M}$ nie erreichen, was zu einer fälschlichen weiteren Anhebung von β führt. In der Regel sind die Aktionswerte allerdings eine schnell veränderliche Weiterverarbeitung von Sensorwerten, wodurch dieser Fall praktisch vernachlässigt werden könnte. Im Falle der Verwendung eines Zahlenformats mit geringer Genauigkeit und sehr rauscharmen Sensorsignalen, ist es denkbar, das durch die Quantisierung häufiger mehrere identische maximale Aktionswerte auftreten. Weiterhin ist bei einer zu stark quantisierten *Reward*-Funktion Vorsicht geboten. Durch eine einfache Addition eines kleinen Betrags auf eines der Maxima kann diese Symmetrie allerdings

kostengünstig gebrochen werden, ohne dabei das Ergebnis maßgeblich zu verfälschen. Ein leichtes Rauschen verhilft auch hier zum Erfolg des Verfahrens.

7.2.2 Verhaltensregulation

Die Regelung von β auf einen Wert, der die Varianz der Auswahlwahrscheinlichkeiten konstant hält, hat im Grunde ausschließlich die Aufgabe zu verhindern, dass die Boltzmann-Selektion in Bereiche driftet, in denen die Entscheidungen entweder nur zufällig oder nur vollständig deterministisch sind. Das vergleichsweise kleine Fenster, in dem eine Regelung über β zu optimalen Verteilungen führt ist dynamisch von den Aktionswerten abhängig und verglichen mit dem für β zulässigen Intervall $(0, \infty)$ verschwindend klein. Durch die Varianzregelung wurde aber ein präziser und zudem recht ausgewogener Regelbereich erzeugt, welcher nun von einer Lernregel verwendet werden kann, um das Verhalten zu beeinflussen.

Addiert man einen leichten *offset* v_0 auf die Varianz so kann nun das Verhalten des Individuums gezielt in Richtung Zufälligkeit oder Determiniertheit gelenkt werden. Beispielsweise wird im Rahmen dieser Arbeit der Aufenthalt in stets demselben Zustand durch eine Anhebung der Zufälligkeit der Aktionen kompensiert. Die Ausführung der ständig gleichen Aktionen, ohne dass sich der Zustand dabei verändert, resultiert im Allgemeinen in sehr eintönigem Verhalten. Möglicherweise erwirtschaftet das Motivationssystem noch geringe Belohnungen durch exzessive Verbesserung ein und derselben Sensor-Aktor-Kopplung. Aus den Beobachtungen zeigt sich aber, dass solche Spezialisierungen beispielsweise auf echter Hardware zu sich schnell erhitzenden Motoren führen können oder den allgemeinen Lernprozess unnötig aufhalten. Unter Verwendung von Temperatursensoren könnte man dem System über den Belohnungsmechanismus negative Rückkopplung geben, wenn seine Aktionen zu erhöhter Temperatur der Motoren führen. Allerdings wurde in den Abschnitten 2.4 und 4.5 vorerst alle extrinsischen Motive ausgeschlossen, weshalb die Temperaturüberwachung außerhalb des Lernalgorithmus stattfindet und diesen ggf. pausiert.

Eine in [7] vorgeschlagene Methode sieht vor, die Regelung der Verhaltensauswahl in Abhängigkeit des gemessenen Prädiktionsfehlers zu gestalten. Ist der Prädiktionsfehler hoch sollte das Individuum die determinierten und bewährten Aktionen aussuchen, sich also konservativer verhalten. Bei einer kontrollierten Situation kann das Individuum durch zunehmend zufälligere Aktionen aber behutsam wieder mehr Exploration wagen.

7.3 Ausübung motorischer Aktionen

Bisher wurde ausführlich beschrieben, in welcher Weise das Individuum aus einer vorgegebenen, noch nicht näher spezifizierten Menge verschiedener Motoraktionen auswählen kann und wie es diese Auswahl bewertet. An dieser Stelle soll nun geklärt werden, welche Aktionen das nun im Speziellen sind. Diese Fragestellung kann prinzipiell nicht isoliert vom Körperbau betrachtet werden. Vielmehr muss nun die Schnittstelle vom Lernalgorithmus zum Körper definiert werden, welche dann abstrakte Handlungsanweisungen in basale Motorbefehle übersetzt. Solch ein *Motormodul* muss also dem physikalischen (oder virtuellen) Körper angepasst werden. Dabei ist es wichtig den

möglichen Aktionsraum *sinnvoll* aufzuteilen, weil hier bisher von einer vergleichsweise geringen Anzahl diskreter Aktionen ausgegangen wird.

Sinnvolle Motoraktionen können dabei sehr unterschiedlicher Natur sein. Obwohl in dieser Arbeit mit basalen Motoraktionen gearbeitet wird, sind auch ganze Bewegungssequenzen wie Motorimpulse oder komplexere Bewegungen als vorgefertigte Aktionen denkbar. Solche *höherwertigen Bewegungsabläufe* werden im Rahmen dieser Arbeit nicht untersucht. Sie erfordern, dass weitere Annahmen über das System gemacht werden, insofern sie nicht bereits selbst durch bootstrapping wie z. B. in [27] entstanden sind. Weiterhin beantwortet die Verwendung solcher Bewegungen nicht die Fragestellung, ob komplexes Verhalten aus ganz basalen Elementen emergieren kann. Daher wird auch hier reduziert. Interessant wäre darüber hinaus auch die Verwendung multimodaler Aktorik wie beispielsweise ein vom Roboter generiertes Piepsen oder Leuchten. Die Anwendung dieser Art Aktuatoren ergibt aber bekanntlich nur einen Sinn für das Individuum, wenn es über seine Sensorik irgendeine Art Reaktion auf diese Signale detektieren kann.

Gegeben ist also eine Liste mit M Einträgen in der nur an der Stelle mit Index i eine 1 und ansonsten Null steht. Der Index i markiert also die vom System zur bevorstehenden Ausführung ausgewählte Aktion. Gesucht ist nun die Abbildung eines binären Aktionsvektors auf einen reellwertigen Motorvektor $\mathbf{m} \in \mathbb{R}^{\Lambda}$. Die Anzahl der Dimensionen des Vektors entspricht dabei der Anzahl der motorischen Freiheitsgrade des Systems. Alle in dieser Arbeit untersuchten Morphologien sind o. B. d. A. jeweils mit zwei Freiheitsgraden ausgestattet. So gehört es vordergründig zum Konzept *vorerst* die Anzahl der Freiheitsgrade gering zu halten, um sowohl den Komplexitätsgrad der Berechnungen also auch Schwierigkeiten bei der Analyse und Visualisierung zu reduzieren. Dabei stellen zwei Freiheitsgrade einen guten Kompromiss aus den Aktionsmöglichkeiten des Systems und dem Aufwand für Berechnungen und Visualisierung dar. Der Wertebereich der Komponenten von $\mathbf{m}$ ist dabei $[-1, +1]$, wobei das Vorzeichen die Richtung und der Betrag die Stärke der Kraftausübung angibt. Dabei steht eine 1 für maximale Kraft bzw. Drehmoment.

7.3.1 Kriterien für basale motorische Aktionen

Für die Anwendung am echten Roboter unterliegen die möglichen motorischen Aktionen besonderen Anforderungen. Besonders wichtig ist es, dass ausgeführte Aktionen den Roboter selbst nicht beschädigen. So selbstverständlich dieser Gedanke ist, so muss doch mit Sorgfalt sichergestellt werden, dass diese Bedingungen gegeben sind. Nicht selten sind die Servomotoren in der Lage Drehmomente aufzubringen welche die mechanischen Endanschläge der Gelenke erheblich herausfordern. Beim Design autonomer mobiler Roboter wird von Seiten der Mechanik ein Kompromiss zwischen dem Leichtbau und der für die Bewegungen notwendigen Kraft gefunden. Dabei ist die Kraft meist ausreichend um bei unangemessen ausgeführten Bewegungen schwerwiegende Schäden an der Robotorplattform zu produzieren. In jedem Fall muss sich der Programmierer in der Pflicht sehen, die Software mit entsprechenden Schutzmechanismen auszustatten, um unnötige Reparaturarbeiten zu vermeiden. Ein zu gedankenlos ausgeführtes

motor babbling[1], d. h. zufällige Bewegungen auszuführen, ist in den seltensten Fällen anwendbar.

Bei der Ausführung des hier vorgestellten unüberwachten Lernverfahrens geht es vordergründlich um Selbstexploration. Im Speziellen bedeutet dies, dass ein Individuum auch die Grenzen seines Bewegungsspielraumes auslotet. In Abschnitt 2.4 wurde die Prämisse gestellt, dass der Selbsterhalt des Individuums vorerst gegeben ist und somit im Algorithmus nicht weiter beachtet wird. Demnach gibt es ausdrücklich *keine Rückmeldung* darüber, ob eine Aktion für den Körper in irgendeiner Art und Weise ungeeignet oder schädlich ist. Ob ein Endanschlag erreicht ist ließe sich nichtsdestotrotz über multimodale Sensoren erfahren. Beispielsweise würde sich beim Erreichen eines Endanschlags der Winkel nicht mehr so rasch ändern, der Stromverbrauch des Servomotors würde sich hingegen drastisch erhöhen und möglicherweise ist eine Spitze in den Beschleunigungsdaten zu messen. Diese Information entsprechend aufbereitet hat viel Potential und mit ihr ist man zu großen Teilen in der Lage eine Selbstverletzung des Roboters zu vermeiden. Außerdem kann man diese Information auch als Lohn oder Strafe verpackt dem Algorithmus als Lernsignal zuführen, sodass z. B. Aktionen die viel Strom verbrauchen, aber wenig bewirken, systematisch vermieden werden. Für die Auswahl der Motoraktionen ist hier wichtig, dass die Stärke des verfügbaren Drehmoments insoweit eingeschränkt wird, dass die Mechanik des Roboters im Mittel nicht übermäßig belastet wird.

An und für sich ist ein Servomotor schon ein komplexes System mit einer Reihe interessanter physikalischer Eigenschaften, welche zum Gesamtsystem dazugehören und auch als solche ihre Daseinsberechtigung haben. Er kann bei weitem nicht auf einen einfachen Drehmomentproduzenten reduziert werden. So spielen Reibung, Anlaufzeiten, interne Verformungen, Getriebespiel und Erwärmung eine entscheidende Rolle im Verhalten des Motors [24]. Der Algorithmus erforscht also nicht nur seinen Körper im Sinne des Aktionsspielraums seiner Gliedmaßen und den Wertebereich seiner Sensorik, sondern exploriert auch zwangsweise die Eigenschaften seiner Motoren. Das Motormodul und die Auswirkungen der von ihm generierten Motoraktionen sind somit auch ein Teil des zu explorierenden Gesamtsystems.

Motoraktionen

Eine motorische Aktion ist im Falle des abstrakten simulierten Testsystems schlicht eine Eingabe, welche über einen Gewichtsvektor eingespeist wird, und somit vergleichsweise unkritisch. Die Auswirkungen der Aktion werden nach der Durchführung des Aktualisierungszyklus sichtbar. In Anwendung auf der echten Roboterplattform wird dem Servomotor ein Steuerbefehl übergeben und dieser bleibt mindestens für die nächsten 10 ms aktiv. Um eine gewünschte Spannung U auf den Motor zu geben, steht der Wertebereich von $0 \ldots 1023$ zur Verfügung. Der reellwertige Motorvektor wird also mit 11 Bit quantisiert, wobei 10 Bit die Spannung angeben und das elfte Bit die Polarität, d. h. die Drehrichtung des Motors unterscheidet. Damit wäre eine Diskretisierung eigentlich schon gegeben, allerdings ist diese Unterteilung für den bisherigen Aufbau des Lernverfahrens viel zu fein. Gewünscht ist eine Reduktion auf unter 30 verschiedene

[1] *to babble*, zu deutsch: brabbeln

Motoraktionen, welche sich aber angemessen über den gesamten Bereich aller motorischen Möglichkeiten verteilen. Die angesteuerten Motorkommandos sind wunschgemäß stetig, um Verschleiß und Temperaturentwicklung gering zu halten. Hektisches Hin- und Herzuckeln der Motoren muss vermieden werden. Es ist also Aufgabe des Motormoduls aus einer diskreten Auswahlliste eine stetige Trajektorie des Motorvektors zu machen.

7.3.2 Ein neuronales Motormodul

In [47] wird eine neuronale Implementation eines Motormoduls beschrieben, welches die hier aufgeführten Anforderungen bewältigt. Beschrieben wird ein *neuronales Feld* in Anordnung eines Rings für Dimension $\Lambda = 2$ des Motorvektors. Das spezielle an neuronalen Feldern ist, dass hier die *Neuronen auch einen Ortsvektor* haben. Der Ort des Neurons kodiert dabei die Stärke der synaptischen Verbindungen und die Verbindungsstruktur. So sind im Falle des Motormoduls alle Neuronen auf einem Ring angeordnet, wobei ihr relativer Abstand zueinander die Stärke der Verbindungsgewichte definiert. Das neuronale Feld ist vollvernetzt. Die Aktivierung der Ringneuronen berechnet sich durch

$$y_j(t) = \tanh\left(\sum_{i=1}^{M} c_{ji} y_i(t-1) + I_j\right), \tag{7.10}$$

wobei I_j die jeweilige Netzeingabe

$$I_j = \begin{cases} I_{Ex} & \text{wenn } a_j = a(t) \\ 0 & \text{sonst} \end{cases} \tag{7.11}$$

aus der gegebenen Auswahlliste M diskreter Aktionen ist. Die Gewichte der synaptischen Verbindungen aller Neuronen innerhalb dieses Rings wird durch

$$c_{ji} = -c_I + c_E\, e^{-\frac{\|\mathbf{r}_j - \mathbf{r}_i\|^2}{2\sigma^2}} \tag{7.12}$$

angegeben, wobei $\mathbf{r}_j$ der Ortsvektor des Neurons j ist. Bei diesem Modul werden die Gewichte voreingestellt, es findet kein Lernen statt. Die Abbildung 7.2 b zeigt die Voreinstellung der Gewichte in Abhängigkeit zum relativen Winkelabstand ausgehend von einem beliebigen Neuron. Die Tabelle 7.1 enthält die Parameter zur Einstellung der Gewichte.

c_E	c_I	I_E	σ
0,7	0,1	0,3	0,5

Tabelle 7.1: Parameter des Motormoduls.

Im Prinzip ist das neuronale Feld so verschaltet, dass die Neuronen lokal erregt werden, wohingegen weiter entfernte Neuronen inhibiert werden. Daraus ergibt sich eine Stabilisierung der Erregung um ihr Zentrum. Ändert sich das Zentrum der Aktivierung, so wandert eine Art Aktivierungswelle entlang des Rings. In der Tat hat die Art der Aktivierungsausbreitung einen Hauch der Eleganz einer bewegten Flüssigkeit. Durch die leichte Selbstkopplung ergeben sich zeitgleich Tiefpasseigenschaften und zu kurze Erregungen führen zu keiner ausgeprägten Aktivierung.

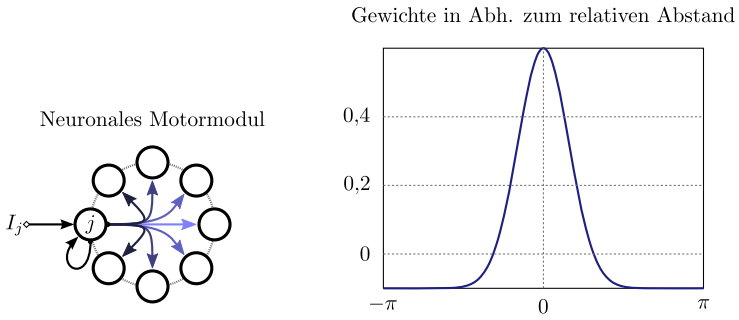

Abbildung 7.2: Neuronales Motormodul. Ausgehende Gewichte eines Neurons j eingezeichnet. Stärke der Verbindungsgewichte in Abhängigkeit zum Abstand.

Verallgemeinerung auf höhere Dimensionen

Der Neuronenring kann auch für höhere Dimensionen des Motorvektors verallgemeinert werden. So bilden die Neuronen für $\Lambda = 3$ eine Kugeloberfläche und allgemein eine $(\Lambda - 1)$-Sphäre. Die Anzahl der verwendeten Neuronen entspricht dabei der Anzahl zu unterscheidender Motoraktivitäten. Dabei offenbart sich wiedereinmal der *Fluch der Dimensionalität*. Für eine geringe Anzahl an Freiheitsgraden ist dieser Ansatz noch praktikabel. Für die Anwendung auf einem *humanoiden* Roboter [17, 18] mit 19 Freiheitsgraden würde das aber schon bei einer räumlichen Auflösung von nur zwei verschiedenen Aktionen pro Freiheitsgrad ein neuronales Feld mit 2^{19} Neuronen erfordern. Demnach liegt es auf der Hand, dass der Ansatz diskreter Aktionen auf diese Weise hier schnell an seine Grenzen stößt.

Implementation und Test

Im Zuge der hier durchgeführten Implementation wurde das Modul zuerst in das in Abschnitt 3.1.1 beschriebene Neuronenmodell übersetzt und die Parameter für die jeweilige Morphologie angepasst. Jedes der Ringneuronen kodiert im Aktionsraum eine bestimmte Richtung in welche eine motorische Aktion ausgeführt werden kann. Die Aktivierung steht dabei für die Stärke der Aktion. Die Ausgänge der Ringneuronen werden gemäß

$$\mathbf{m}(t) = \sum_{k=1}^{M} y_k(t) \begin{pmatrix} \cos(\varphi_k) \\ \sin(\varphi_k) \end{pmatrix} \tag{7.13}$$

verrechnet und ergeben zusammen den Motorvektor $\mathbf{m}(t)$, wobei $\varphi_k = 2\pi k/M$ ist. Das gesamte Modul ist in Abbildung 7.2 a zu sehen. Um die Funktion zu demonstrieren wurde exemplarisch ein kurzes Stück des Verlaufs einer üblichen Aktionsauswahl für $M = 30$ diskrete Aktionen und $\Lambda = 2$ Dimensionen des Motorvektors abgetragen. Die Aktivierungen auf dem Motorring sowie das motorische Ausgangssignal sind in Abbildung 7.3 dargestellt. Für die Analyse wurde ein aussagekräftiger Ausschnitt eines repräsentativen Experiments mit einer Länge von 50 Zeitschritten herausgegriffen. Die obere Abbildung zeigt die von der Aktionsauswahl angewiesene Aktion I_j. Hinterlegt sind die Aktivierungen der Ringneuronen zu jedem Zeitschritt, wobei eine rote Färbung eine stark positive Aktivierung und eine blaue Färbung eine stark negative Ak-

tivierung bedeutet. Zur Unterscheidbarkeit einzelner Aktionen sind die Ringneuronen durch ihre Winkel (d. h. durch Positionen auf dem Ring) gekennzeichnet. Im Verlauf

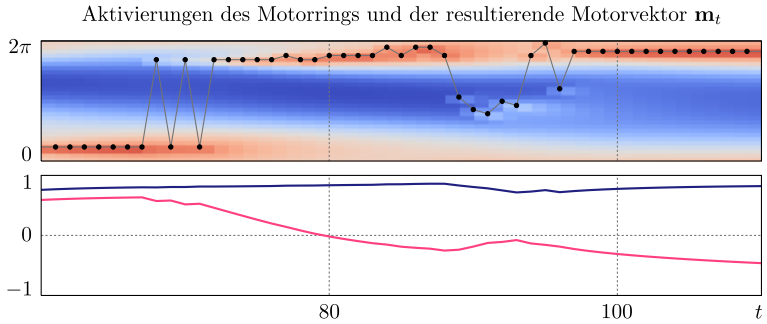

Abbildung 7.3: Verlaufs einer üblichen Aktionsauswahl für $M = 30$ diskrete Aktionen und Dimension $\Lambda = 2$ des Motorvektors $\mathbf{m}(t)$. Oben ist die diskrete Auswahl der Aktionen hinterlegt mit den Aktivierungen des neuronalen Feldes. Unten sind die Komponenten des resultierenden Motorvektors abgebildet.

des Experiments wechselt die Aktion mehrfach. Dabei ist zu beobachten, dass ein kurzes unentschlossenes Hin- und Herwechseln keine sprunghaften Auswirkungen auf den Motorvektor hat (Tiefpasswirkung). Nachdem die Aktion nun endgültig gewechselt hat verschieben sich die Aktivierungen in dem neuronalen Feld und der Motorvektor ändert sich. Nach einigen Zeitschritten testet das Aktionsauswahlmodul einige neue Aktionen an, was zu einer geringen Änderung des Motorvektors führt, aber kehrt dann wieder zu der vorherigen Aktion zurück.

Zusammenfassung

Damit der Lernfortschritt gerichtet ist, werden Zustände und Aktionen bewertet. Dazu wird die SARSA(λ)-Lernregel aus der Familie des Bestärkendes Lernens verwendet. Die Auswahl der motorischen Aktionen erfolgt auf Grundlage dieser Bewertung durch eine Boltzmann-Selektion, welche unter Regelung der inversen Temperatur (in Abhängigkeit der Varianz) immer möglichst ausgewogen ist. Dabei ist es möglich das Verhalten zu regulieren, indem die Aktionsauswahl entweder zufälliger oder determinierter ausfallen kann. Die diskrete Aktionsauswahl wird dann mit Hilfe eines neuronalen Motormoduls in kontiniuierliche Motorwerte übersetzt.

8 Implementation, Experimente und Auswertung

In den vorangegangenen Kapiteln wurde das Modell des Individuums beschrieben und erläutert, wie die einzelnen funktionalen Einheiten implementiert werden. Da nun alle notwendigen Bestandteile des Systems vorliegen, muss geklärt werden, welche Morphologien Körper und Umwelt bilden, und das Gesamtsystem getestet werden. Auf der Basis eines funktionierenden Gesamtsystems können nun die Untersuchungen durchgeführt werden. Dieses Kapitel ist folgendermaßen strukturiert: Zu Beginn werden die, für die Experimente verwendeten, Morphologien vorgestellt. Daraufhin wird das in Kapitel 4 beschriebene Modell nochmals aufgegriffen, aber diesmal im Überblick mit den bestehenden Komponenten betrachtet. Darauf folgen Details zur Implementation, eine Bewertung der Rechenzeit- und Speicherkomplexität und die Durchführung eines Funktionstests des Gesamtsystems. Im dritten Abschnitt wird der Aufbau und die Durchführung der Experimente erläutert. Das Kapitel endet mit der Auswertung der Experimente.

8.1 Beschreibung der Morphologien

8.1.1 Abstrakte Miniaturwelten

Die Morphologien für den Test des Algorithmus sollen beschränkte Systeme sein, welche wohldefiniert sind und zeitdiskrete, wertkontinuierliche Sensordaten produzieren sowie Motordaten verarbeiten können. Sie sollen von niedriger Dimension sein, somit sei die Welt auf zwei sensorische und zwei motorische Dimensionen reduziert. Die sensorische Dimension wird darauf, wie bereits erwähnt, um die vergangenen motorischen Werte, die zeitverzögerten Kopien und den Bias erweitert.

Für die Simulation sollte die Berechnung der dynamischen Welt nicht allzu aufwändig sein. Dazu bietet es sich an als Körper und Umwelt selbst ein Zwei-Neuronen-Netz zu verwenden. Es ist auf das Intervall $(-1, 1)$ beschränkt, leicht zu berechnen und bietet durch zahlreiche Vernetzungsmöglichkeiten viel Auswahl an verschiedenen Dynamiken. Die Welt wird mit

$$\mathbf{x}(t+1) = \tanh\left(\tilde{\mathbf{W}}\mathbf{x}(t) + \tilde{\mathbf{b}} + \tilde{a}\mathbf{m}(t)\right) \tag{8.1}$$

berechnet. $\mathbf{x}(t)$ ist der Zustandsvektor der Welt und gleichermaßen die *sensorische Umweltinformation* für das Individuum. $\mathbf{m}(t)$ sind demnach die zu schreibenden Motorwerte. Die Gewichtsmatrix $\tilde{\mathbf{W}}$ und der Biasvektor $\tilde{\mathbf{b}}$ sind die Parameter der Welt. Der Inputvektor $\mathbf{m}(t) \in [-1, 1]$ wird bei allen Miniaturwelten mit $\tilde{a} = 0{,}1$ skaliert. Es wurden insgesamt drei verschiedene Morphologien ausgewählt (vergleiche dazu Abbildung 8.1).

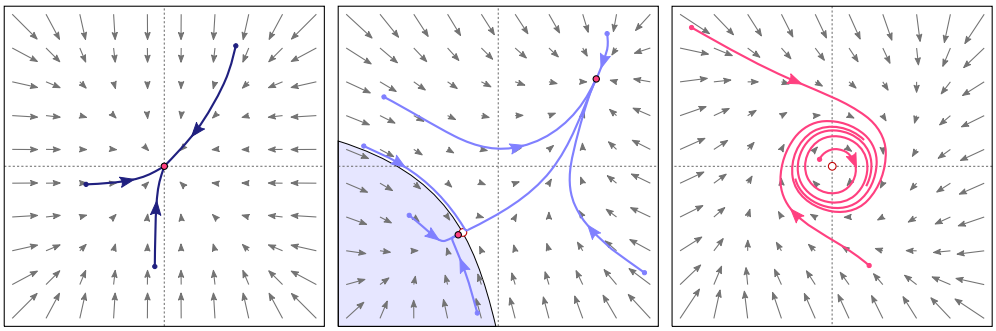

Abbildung 8.1: Drei verschiedene Morphologien zur Erprobung des Lernverfahrens. Alle Welten sind auf $(-1, 1)$ beschränkt. Die Vektorpfeile (normiert) kennzeichnen die Stärke und Richtung in der die Dynamik der Welt verläuft. Zur Illustration sind Trajektorien zu verschiedenen Startpunkten eingezeichnet. Rote Punkte kennzeichnen die stabilen, kleine rote Kreise die instabilen Fixpunkte.

Die Parameter der drei Welten sind gegeben als:

$$\tilde{\mathbf{W}}_{\#1} = \begin{pmatrix} 0{,}99 & 0 \\ 0 & 0{,}99 \end{pmatrix} \qquad \tilde{\mathbf{b}}_{\#1} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{8.2}$$

$$\tilde{\mathbf{W}}_{\#2} = \begin{pmatrix} 1{,}01 & 0{,}1 \\ 0{,}1 & 1{,}01 \end{pmatrix} \qquad \tilde{\mathbf{b}}_{\#2} = \begin{pmatrix} 0{,}0398 \\ 0 \end{pmatrix} \tag{8.3}$$

$$\tilde{\mathbf{W}}_{\#3} = \begin{pmatrix} 1{,}01 & 0{,}1 \\ -0{,}1 & 1{,}01 \end{pmatrix} \qquad \tilde{\mathbf{b}}_{\#3} = \begin{pmatrix} 0{,}001 \\ 0{,}001 \end{pmatrix} \tag{8.4}$$

Die erste Welt ist vergleichbar mit einer Mulde. In Abwesenheit einer Kraftanstrengung rutscht das Individuum immer weiter in die Mitte und kommt schließlich im Zentrum zur Ruhe. Dort befindet sich ein Punkt an dem die Dynamik zum erliegen kommt, ein sogenannter *stabiler Fixpunkt*. In den Randbereichen ist das Vektorfeld zu stark, als dass das Individuum diese Grenze überschreiten könnte. Das ist zugleich eine Gemeinsamkeit aller drei Miniaturwelten. Sie sind jeweils auf den Bereich ± 1 beschränkt und bilden somit abgeschlossene Welten. Welt Nr. 2 ist sehr ähnlich aufgebaut, allerdings besitzt diese Welt einen weiteren stabilen Fixpunkt. Der eine liegt bei $\mathbf{x}_1^* \approx (0{,}61\ 0{,}55)^T$ der andere bei $\mathbf{x}_2^* \approx (-0{,}25\ -0{,}43)^T$. Stabile Fixpunkte sind *Attraktoren*, sie haben eine anziehende Wirkung. Die beiden stabilen Fixpunkte haben unterschiedlich starke Einzugsbereiche, sogenannte *Basins*. Die Grenze zwischen den Basins, die *Separatrix*, verläuft ganz in der Nähe des zweiten Fixpunkts, sodass der Weg aus dem ersten Fixpunkt sehr lang ist, bis das Basin verlassen werden kann. Bildlich vorzustellen ist diese Welt wie ein Tal und ein Berg mit einem kleinen Plateau zum Rasten. Die dritte Welt hat gegenüber den anderen beiden eine Besonderheit. Sie hat keine stabilen Fixpunkte, nur einen *instabilen*. Dieser liegt im Zentrum und treibt nach außen. Das Vektorfeld ist hierbei wie ein Karussell. Ohne Kraftanstrengung wird das Individuum im Kreis gewirbelt (Uhrzeigersinn). Der stabile Zustand ist die Oszillation, ein sogenannter *quasi-periodischer Attraktor*. Befindet man sich außerhalb

wird man ebenfalls vom Attraktor angezogen (vgl. dazu [33]). Alle drei Welten sind zusammenfassend in Abbildung 8.1 gegenübergestellt.

8.1.2 Die Roboterplattform SEMNI

Semni wurde im Labor für Neurorobotik (NRL) an der Humboldt-Universität zu Berlin entwickelt und speziell für die Anwendungen unüberwachter Lernverfahren konzipiert. Das ägyptische Wort bedeutet *sich etablieren*, wobei das Akronym Semni auch für *Selbst-explorierendes multi-neurales Individuum* steht. Die Konstruktion ist so ausgelegt, dass sich der Roboter bei der Ausführung seiner motorischen Aktionen nicht selbst beschädigen kann, aber dafür reichhaltige sensorische (speziell propriozeptive) Rückmeldungen seines Körpers erhält. Semni hat nur zwei Freiheitsgrade in Form zweier

Abbildung 8.2: Semni: Ein Roboter für die Erprobung unüberwachter Lernverfahren zur Selbstexploration.

Servomotoren, welche die Hüfte und das Knie bilden. Die Anzahl der Freiheitsgrade wurde bei dieser Roboterplattform bewusst gering gehalten, um die Analyse der Ergebnisse weitestgehend zu vereinfachen. Der Bewegungsspielraum von Semni bleibt bedingt durch die Art der Anbringung der Motoren auf die Sagittalebene beschränkt. Allerdings stehen dem Roboter innerhalb dieser Ebene vielfältige Bewegungsmöglichkeiten zur Verfügung. Mögliche Positionen und Bewegungsmodi sind z. B. das Liegen auf Bauch und Rücken, Stehen in mehreren stabilen Zuständen sowie Wippen und Überrollen. Unter Einsatz der vollen Motorleistung ist für Semni sogar ein Salto ausführbar.

Besonders erwähnenswert ist die reichhaltige sensorische Ausstattung des Roboters. Neben den obligatorischen Drehgebern in Hüfte und Knie stehen drei orthogonale Beschleunigungswerte zur Verfügung, welche im Kopf ermittelt werden. Weiterhin werden jeweils Messwerte der Stromaufnahme und Innenraumtemperatur der Servomotoren zur Auswertung bereitgestellt. Optional besitzt Semni einen Abstandssensor, welcher am Kopf befestigt und nach vorn heraus gerichtet ist.

Für die Untersuchungen mit dem Roboter werden die Winkelstellungen der Gelenke (Hüfte und Knie), die Stromaufnahme der jeweiligen Servomotoren sowie die zwei Beschleunigungskomponenten innerhalb der Sagittalebene verwendet. Die Sensordaten setzen sich demnach wie folgt zusammen:

$$\mathbf{x}(t) = (p_h(t), p_k(t), I_h(t), I_k(t), a_{\leftrightarrow}, a_{\updownarrow})^T \tag{8.5}$$

Die motorischen Aktionen, welche SEMNI ausführen kann, sind die Ansteuerung der Spannung der Servomotoren unter Angabe von Betrag und Richtung sowie das Einschalten eines Entspannungsmodus für die Motoren. Für die optische Ausgabe kann SEMNI die Helligkeit zweier Leuchtdioden regulieren.

Die Übertragung der Daten findet im garantierten Takt von 100 Hz statt. Neben dem seriellen Bus zur Kommunikation wird der Roboter über zwei weitere Leitungen mit Spannung versorgt, welche SEMNI ebenfalls als Sensorwert bereitstellt. Die Roboterplattform SEMNI ist explizit nicht-autonom. Der Roboter ist für die Langzeiterprobung unüberwachter Lernverfahren ausgelegt, weshalb das Lernen und die Steuerung auf eine externe Maschine ausgelagert wird. Das vereinfacht die Programmierung und bietet Raum zum experimentieren. Und weil für viele Anwendungen eine gute Visualisierung essentiell ist, stehen mit dem Werkzeug PC dem Experimentator viele Möglichkeiten für die Anzeige von Sensordaten und Lernsignale zur Verfügung. Über ein Erweiterungsmodul kann SEMNI außerdem auch an die NRL-Software *BrainDesigner*[1] angeschlossen werden. In einer kommenden Ausbaustufe wird SEMNI um weitere spezielle taktile Sensoren erweitert, um Berührungen seines Körpers zu detektieren.

8.2 Inbetriebnahme des Gesamtsystems

Nachdem nun verschiedene Morphologien als mögliche Testszenarien beschrieben wurden, kann das Gesamtsystem in Betrieb genommen werden. Abbildung 8.3 zeigt noch einmal zusammenfassend das Modell des Individuums unter Einbindung der im bisherigen Verlauf der Arbeit beschriebenen Module.

8.2.1 Implementation der Experimentierumgebung

Als Grundlage der Experimente wurden für die abstrakten Morphologien und die Roboterplattform jeweils eine Experimentierumgebung programmiert und der Algorithmus darin implementiert. Die Experimentiersoftware ist in der Programmiersprache C geschrieben und kann durch die Verwendung des *Simple DirectMedia Layer* (SDL) unabhängig vom Betriebssystem übersetzt werden. Für eine performante grafische Ausgabe der zahlreichen Systemvariablen findet die *Open Graphics Library* (OpenGL) Verwendung. Über die grafische Ausgabe können die verschiedenen Sensor- und Motorsignale, das Wachstum des Expertengases, Fehler- und Nützlichkeitswerte, die Q-Matrix und die Aktionsauswahl überwacht werden. Die simulierten Morphologien können dabei nach Belieben in Zeitlupe, Echtzeit oder im Zeitraffer betrachtet werden. Für die störungsfreie Kommunikation mit dem Roboter wurde das Auslesen und Schreiben auf

[1] *BrainDesigner* ist eine Software zur manuellen Verschaltung und Visualisierung von künstlichen neuronalen Netzen. Sie wird von Christian Thiele am Labor für Neurorobotik entwickelt.

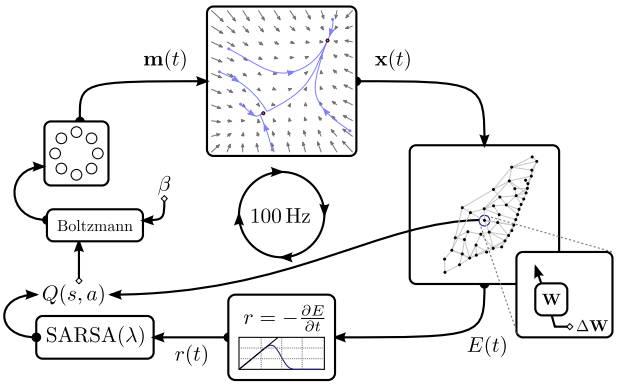

Abbildung 8.3: Übersicht: Gesamtarchitektur mit Komponenten

den seriellen *com port* in einen separaten *thread* ausgelagert. In beiden Experimentierumgebungen kann mittels eines *control pads* in das Geschehen eingegriffen werden, um beispielsweise manuell den Aktionsraum der jeweiligen Morphologie zu erproben.

Details zur Implementation

Der Aktionsraum der abstrakten Testsysteme und von Semni musste drastisch reduziert werden. Die anfänglich gewünschten dreißig diskreten Aktionen wurden auf nur noch 8+1 verschiedene Motoraktionen zusammengekürzt, wobei eine Aktion *tue nichts* ist. Der tabellarische Aufbau der Aktionsbewertung enthält bisher keinerlei Generalisierungsfähigkeit. Daher muss jede Aktion in jedem Zustand separat erprobt werden, was das Lernen ungemein verzögert und erschwert. Bei einer höheren Auflösung der Aktionsauswahl haben viele Aktionen aber eine ähnliche, wenn nicht exakt gleiche Wirkung, daher sollte für fortgesetzte Untersuchungen der tabellarische Ansatz verworfen und ein kontinuierliches Aktionsspektrum erprobt werden.

Das Filter für die Ableitung des Fehlersignals als Belohnungssignal hat eine nicht vernachlässigbare Verzögerungszeit von $t_D = 10$ Zeitschritten. Das muss bei der Bewertung der Zustände berücksichtigt werden. Dazu werden die Zustands-Aktions-Paare des jeweiligen Zeitschritts ebenfalls um t_D verzögert, um somit im richtigen zeitlichen Kontext in der Lernregel (7.1) verrechnet zu werden.

Die Filterung wird auf jedem der N separaten Fehlersignale durchgeführt und nach der Bestimmung des Gewinnerexperten die Belohnung entsprechend berechnet. Denkbar wäre es auch nur das Gesamtfehlersignal, zusammengesetzt aus den Fehlersignalen des jeweils besten Experten, zu verwenden und dieses dann abzuleiten. Das wäre wesentlich effizienter, allerdings enthält das zusammengesetzte Signal Diskontinuitäten, welche bei einer Ableitung zudem noch verstärkt würden. Darüber hinaus fügt es weitere Information hinzu. Somit würde das Umschalten von einem schlechteren auf einen besseren Experten zusätzlich belohnt werden. Im Zuge dieser Arbeit wurden die Auswirkungen dessen bisher aber nicht im Detail untersucht.

8.2.2 Abschätzung der Rechenzeit- und Speicherressourcen

Die Anzahl N der Experten wächst über die Zeit von anfänglich zwei auf höchstens N_{max}. Diese obere Schranke kann entweder im Hinblick auf die zugrundeliegende Rechenarchitektur eingestellt werden, um die vorgegebene Rechenzeit einzuhalten oder kann manuell an die erwartete Schwere des Problems angepasst werden. Drei weitere Größen sind dabei von entscheidender Bedeutung für die Komplexität des gesamten Algorithmus.

Die erste ist die Dimension D des sensomotorischen Zustandsvektors, d. h. die Anzahl der verwendeten Sensoreingaben. In der Praxis kann der Aufwand reduziert werden, indem nicht der gesamte Zustandsvektor $\mathbf{x}(t)$ geschätzt wird, sondern nur einige Komponenten davon. Diese besteht dann aus einer Auswahl besonders aussagekräftiger Sensorkanäle, sodass $\tilde{D}$ die Dimension einer reduzierten Schätzung mit $\tilde{D} < D$ ist. Möglicherweise hilft hier eine Hauptkomponentenanalyse, um die Auswahl zu erleichtern. Die zweite wichtige Größe ist die Dimension K der expliziten zeitlichen Einbettung. Die Dimension des sensorischen Zustandsvektors vergrößert sich von D auf $KD + 1$. Eine implizite zeitliche Einbettung würde zwar die Dimension der Eingabe verringern, erhöht im Allgemeinen aber auch die Komplexität der Verarbeitung, z. B. durch zusätzliche verdeckte Neuronen (vergleiche Abschnitt 5.1.5). Die dritte Größe ist die diskrete Anzahl M verschiedener Motoraktionen. Der eigentliche Motorvektor hat in dieser Arbeit bei allen Anwendungen die Dimension 2 und fällt vorerst aus der Diskussion heraus. Für höhere motorische Dimensionen wurden bereits in Abschnitt 7.3.2 Bedenken bzgl. der Skalierbarkeit geäußert.

Ein separater Experte, hier durch einen mehrdimensionalen FIR-Prädiktor implementiert, hat eine Rechenzeit- und Speicherkomplexität von $\mathcal{O}(KD^2 + D)$ (ohne die o. g. Reduktion). Die obere Schranke wird durch die verwendeten Synapsen gesetzt. Das ist bei neuronalen Architekturen üblich und durch den ausschließlichen Einsatz von Vorwärtsverknüpfungen im Vergleich zu rekurrenten Netzen gering in der Anzahl der Synapsen. Die Parameter K und D sind über die Laufzeit des Algorithmus verglichen mit der wachsenden Anzahl der Experten klein, und vor allem konstant. Für die Betrachtung des gesamten Algorithmus wird daher der Ressourcenverbrauch eines einzelnen Experten als konstant angenähert.

Jeder Experte errechnet nun eine Schätzung der zukünftigen Werte, wobei der Fehler der Schätzung aller Experten miteinander verglichen werden muss, um das Minimum zu identifizieren. Die benötigen Ressourcen dieses Prozesses wachsen also mit $\mathcal{O}(N)$. Der Fehler der Schätzung soll dem System als Lernsignal zur Verfügung gestellt werden. Dazu wird der Fehler wie in Kapitel 6 beschrieben durch ein Tiefpass-Differentiator-Filter abgeleitet. Die Ableitung wird dabei für jedes der Fehlersignale der jeweiligen Experten durchgeführt. Die Kosten der Filterung skalieren mit der Anzahl der Filter-Koeffizienten, welche unter Berücksichtigung der Qualität der Ableitung und der sich für das zu filternde Signal ergebende maximalen Zeitverzögerung optimiert wurde. Die Anzahl der Koeffizienten ist dabei konstant, sodass sich pro durchgeführter Ableitung ebenfalls konstante Kosten ergeben. Dadurch skaliert der Gesamtverbrauch des Algorithmus linear in Abhängigkeit zur Anzahl der Experten.

Die Bewertung der Aktionen hat durch den tabellarischen Charakter einen Speicherverbrauch von $\mathcal{O}(MN)$. Die Aktualisierung der Zustands-Aktions-Matrix verbraucht

im einfachsten Fall eine konstante Laufzeit, da nur ein Feld aktualisiert wird. Durch die Verwendung der *eligibility traces* skaliert dann aber auch die Laufzeit mit $\mathcal{O}(MN)$. Dies ist zwar im speziellen ungünstig, aber es existieren Methoden [45], um die Kosten etwas zu reduzieren. Die Auswahl der als nächstes auszuübenden motorischen Aktionen skaliert linear in der Anzahl der diskreten Motoraktionen, da außer bei einer zufälligen Auswahl alle Einträge $Q(a_i, s)$ mit $i = 1..M$ angefasst werden müssen. Die Bewertung wird also für alle Experten und die Auswahl nur für den Gewinnerexperten ausgeführt. Daher beeinflusst es die Gesamtabschätzung nicht dramatisch und der Laufzeit- und Speicherzuwachs skaliert nach wie vor mit $\mathcal{O}(N)$.

Kritisch betrachtet sagt dies allerdings noch nichts über die Echtzeitfähigkeit des Gesamtsystems aus. Mit Ausnahme der Lösch- und Einfügeoperation für Expertenmodule werden alle Berechnungen der Module zu jedem Zeitschritt erneut durchgeführt. Das heißt die Art und Anzahl der verwendeten Rechenoperationen skaliert ebenfalls linear mit N und kann für ein gegebenes N_{max} abgezählt werden. Für eine effiziente Reduktion der Rechenzeit wird die Verwendung stückweise linear approximierter Varianten des Tangens Hyperbolicus und der Exponentialfunktion empfohlen. Inwieweit die Genauigkeit des für reelle Zahlen verwendeten Zahlenformats reduziert werden kann, muss im Detail untersucht werden und wird hier nicht pauschal beantwortet. Für die Versuche in dieser Arbeit wurden bisweilen für reelle Zahlen ausnahmslos Gleitkommaformate mit doppelter Genauigkeit (64 Bit) verwendet.

8.2.3 Funktionstest des Gesamtsystems

Um die korrekte Funktionsweise des Gesamtsystems zu testen wurde in Anlehnung an klassische Testfälle eine Balancieraufgabe gestellt. Dazu wurde das Erreichen und Stabilisieren eines instabilen Zustands belohnt. Die Aufgabe für das abstrakte Individuum (Miniaturwelt Nr. 3) war die Stabilisierung der Oszillation im Zentrum des Vektorfeldes. Es wurde eine kleine Belohnung für die Annäherung an die instabile Lage gegeben und eine höhere Belohnung, wenn diese auch erreicht wurde und dabei kein hoher motorischer Aufwand mehr nötig ist, um diese Position zu halten. Die letztere Bedingung soll verhindern, dass das System nur durch den instabilen Punkt hindurchfährt. Die Belohnungsfunktionen für Welt Nr. 3 ist

$$r_{\#3}(t) = \begin{cases} 1 & \text{wenn } \|\mathbf{x}(t)\| < 0{,}1 \text{ und } \|\mathbf{m}(t)\| < 0{,}1 \\ -\frac{\mathrm{d}\|\mathbf{x}\|}{\mathrm{dt}} & \text{sonst} \end{cases} \tag{8.6}$$

Das Ergebnis des Versuchs ist in der Abbildung 8.4 dargestellt. Der Vektor $\mathbf{x}(t)$ kennzeichnet wie gehabt die Sensorinformation und entspricht der Position des Individuums, $\mathbf{m}(t)$ sind die Ausgaben des Motormoduls und $r(t)$ die erhaltene Belohnung zum Zeitschritt t. Dem Individuum wurde etwas Zeit gegeben, um sich seinen Zustandsraum aufzubauen und verschiedene Aktionen auszuprobieren. Dabei konnte schon kurz nach Beginn eine Konzentration auf das Zentrum des Vektorfelds beobachtet werden. Bei Zeitschritt 2200 wird die homöostatische Regelung der Aktionsauswahl abgestellt und die Zufälligkeit der Aktionsauswahl Stück um Stück reduziert, d. h. die Aktionsauswahl ist ab diesem Zeitpunkt auf volle Ausnutzung eingestellt und exploriert nicht mehr. Kurz darauf sind zunehmend verlangsamte Nulldurchgänge zu beobachten und

ab dem Zeitschritt 2600 gelingt die Stabilisierung schon recht gut. Das Individuum lässt sich dabei entspannt aus dem Zentrum treiben und sammelt Belohnung. Sobald dann das Verlassen des Zentrums durch einen Experten detektiert wird, steuert es wieder zielgerichtet das Zentrum an.

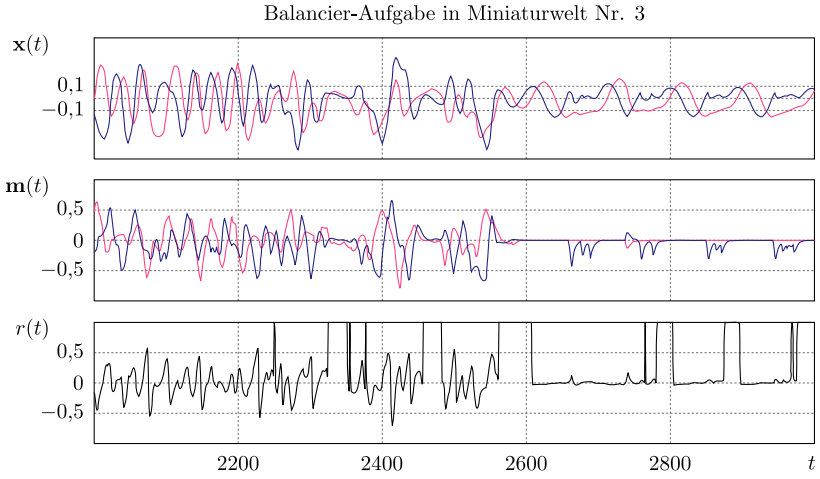

Abbildung 8.4: Balancieraufgabe für das Individuum in Welt Nr. 3: Nach einer kurzen Aufbauzeit für den Zustandsraum wird sukzessive die Zufälligkeit der Aktionsauswahl verringert und es stellt sich bald darauf das gewünschte, um die instabile Lage balancierende, Verhalten ein.

8.3 Beschreibung und Durchführung der Experimente

Mit der Durchführung der Experimente sollen folgende Fragestellungen geklärt werden: Wie unterscheidet sich intrinsisch motiviertes Verhalten von zufälligem Verhalten? Welche Auswirkungen hat diese Unterscheidung auf das Wachstum der Multi-Experten-Struktur und wie entwickelt sich der Vorhersagefehler des Systems? Wie wird der sensomotorische Zustandsraum exploriert und welche Auswirkungen auf die Aktionsauswahl können beobachtet werden?

8.3.1 Experimentalaufbau

Für die Experimente wurden alle Parameter konstant gehalten (siehe dazu Tabellen 8.1 und 5.1) und nur das Belohnungssignal variiert. Dieses wurde im Vergleich zur intrinsischen Motivation durch ein angepasstes Rauschsignal ersetzt. Wenn die Belohnung rein zufällig ist, kann keine kausale Bewertung der Aktionen erfolgen. Nichtsdestotrotz enthält das Gesamtsystem genügend Annahmen, um trotzdem zuverlässig zu funktionieren. Das Verhalten ist zwar nicht mehr im klassischen Sinne motiviert, doch ist es auch nicht vollständig zufällig. Zum Beispiel wechselt nicht zwangsweise in jedem Zeitschritt die Aktion und auch die Auswahl der Aktionen eines jeweiligen Experten ändert sich bedingt durch die Lernrate und Varianzregulation verglichen mit dem Zeittakt verhältnismäßig langsam. Kurz: Das Verhalten ist zwar *zufällig motiviert* aber an

sich schon durchaus brauchbar. Es ist nicht mit einer reinen Zufallsansteuerung der Motoren vergleichbar. In [47] wird die ausgewählte Aktion nur mit einer Wahrscheinlichkeit von 20% gewechselt. Das zufällig motivierte Verhalten des Individuums ist in erster Näherung damit vergleichbar. Ein rein zufälliges Signal auf dem Motorvektor, ohne weitere Maßnahmen, wie einer Tiefpassfilterung, führt in der Regel zu vollständig unbrauchbarem Verhalten.

Das Verhalten des Individuums wird unter Variation der Morphologie, d. h. an den drei verschiedenen Miniaturwelten und dem Roboter Semni untersucht. Alle Experimente sind auf die Dauer von 10^5 Zeitschritten ausgelegt. Das entspricht etwa 17 Minuten bei einer Taktrate von 100 Hz. Eine Ausnahme bildet das Experiment mit Semni. Dieses wird über die Dauer von einer Stunde durchgeführt. Bei Semni wird außerdem die Anzahl der Experten auf 300 erhöht, da der sensorische Zustandsraum wesentlich größer ist. Zum Aufbau der Multi-Experten-Struktur und für die Zustandsidentifiktation wird im Fall der abstrakten Miniaturwelten der zweidimensionale Zustandsvektor (8.1) verwendet. Auf der Roboterplattform stehen nach Definition (8.5) sechs Sensorwerte zur Verfügung. Das System soll davon aber nur die vier Komponenten $p_h(t)$, $p_k(t)$, $a_{\leftrightarrow}(t)$ und $a_{\updownarrow}(t)$ vorhersagen.

Der Lernprozess hat zwar per Definition kein Ende, sehr wohl aber einen Anfang. Zu Beginn müssen demnach einige Startbedingungen definiert werden. Alle synaptischen Gewichte werden zu Beginn zufällig aus dem Intervall $[-w_S, w_S]$ mit $w_S = 10^{-10}$ initialisiert, ebenso die Aktionswerte in der Q-Matrix und die Verzögerungsketten für die Sensorwerte. Die Werte für die inverse Temperatur β werden mit 1 initialisiert, womit die Aktionsauswahl mit einer Gleichverteilung beginnt, bis sie sich durch die Varianzregelung angepasst hat. Es wird klassisch mit zwei Experten begonnen.

N_{max}	M	K	α	γ	λ
50	$8+1$	5	0,5	0,99	0,9

Tabelle 8.1: Standard-Parameter für die Verhaltensexperimente.

8.4 Auswertung der Experimente

Zur Unterscheidung der Individuen wird in Anlehnung an [9] die Notation ψ (Psi) eingeführt, wobei der hochgestellte Index die Morphologie angibt. Die Indices $1, 2, 3$ stehen für die jeweiligen Miniaturwelten und S für Semni. Der tiefgestellte Index gibt an, ob es sich um die intrinsische Motivation (im) oder die zufällige Motivation (rnd) handelt.

Aufenthalt im Zustandsraum

Miniaturwelt 1 hat die einfachste Struktur. Der Zustandsraum ist aber wegen der geringen motorischen Einkopplungsstärke in den Randbereichen nicht erreichbar, da das Vektorfeld zu stark ist. Für beide Motivationssysteme ist der Innenbereich dieser Welt jedoch einfach zu explorieren. Bei dem Verhalten von ψ^1_{rnd} fällt auf, dass die Randbereiche (d. h. die Ecken und Kanten) des noch erreichbaren Zustandsraums wesentlich

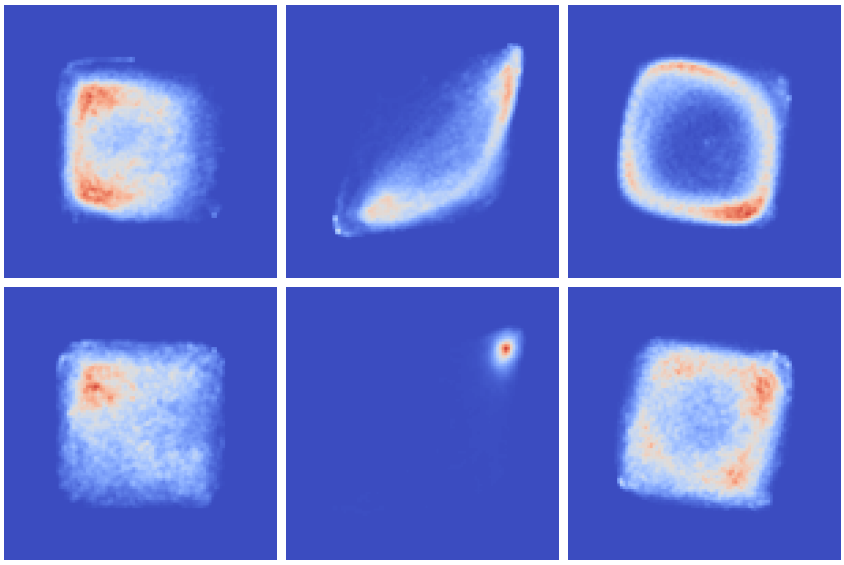

Abbildung 8.5: Häufigkeiten des Aufenthalts der sechs Individuen im jeweiligen Zustandsraum; Miniaturwelt 1:links, 2:mittig und 3:rechts. Intrinsische Motivation oben und zufällige Motivation unten. Blau: keine Anwesenheit, rot: häufige Anwesenheit.

intensiver besucht wurden (vgl. dazu Abbildung 8.5). Der Randbereich kann durch die Nichtlinearitäten schlechter vorhergesagt werden, daher vermeidet ψ^1_{im} den extremen Randbereich. Der Aufenthalt im Zustandsraum ist anfänglich bei beiden Individuen ziemlich ähnlich und nicht signifikant unterscheidbar. Gegen Ende des Versuchs zeigt sich jedoch ein interessanter Effekt. Obwohl die Struktur des Vektorfelds ins Zentrum gerichtet ist ergibt sich eine quasizyklische Verhaltenssequenz. Wie in Abbildung 8.5 sich schon andeutet und in Abbildung 8.6 deutlich sichtbar ist, ergeben sich kurzzeitig stabile Oszillationen, in großem Radius um das Zentrum deren Form dabei kontinuierlich variiert. Die Sequenz besteht dabei aus eine ganzen Reihe von Experten.

Die zweite Morphologie ist schwieriger gestaltet. Die motorische Einkopplung ist verglichen mit dem Vektorfeld zu schwach, um dass ein hektisches Motorzucken zum langfristigen Verlassen des Fixpunkts $\mathbf{x}^*_1$ führen würde. Es muss, ob gezielt oder zufällig, über mehrere Zeitschritte hinweg eine bestimmte Folge von Aktionen ausgeführt werden, um den Fixpunkt zu verlassen und dann möglichst wenig motorische Aktion ausgeübt werden, um nicht wieder hineinzulaufen. Dabei hat sich gezeigt, dass ψ^2_{rnd} es nur äußerst selten schafft den Fixpunkt zu verlassen. Dementsprechend ist sein Aufenthalt besonders in der Nähe dieses Fixpunkts sehr häufig. Der Rest des Zustandsraums wurde nur spärlich oder gar nicht besucht. Im Gegenteil dazu findet ψ^2_{im} den Pfad von Fixpunkt $\mathbf{x}^*_1$ zu $\mathbf{x}^*_2$ und lässt sich mehrfach wieder zurückfallen. Es hält sich vergleichsweise lange in beiden Fixpunkten auf und nutzt anscheinend die Eigenheiten der Morphologie aus. Es bewegt sich dabei häufig auf den effizienten Pfaden. Dabei ergibt sich erneut ein emergenter Effekt aus dem Zusammenspiel von Expertengas und Motivationssystem, indem sich eine andauernde Sequenz aus dem wechselseitigen Besuch der beiden Fixpunkte ausprägt.

Die dritte Morphologie wird von ψ^3_{rnd} ebenfalls gleichmäßig abgesucht. Das Vektor-

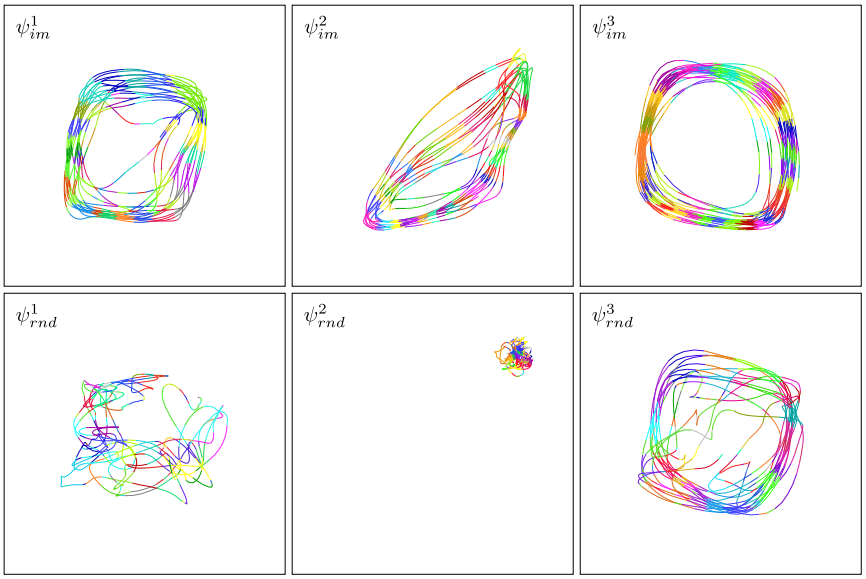

Abbildung 8.6: Verhalten: Abgebildet sind die Trajektorien der verschiedenen Individuen über die letzten 1000 Zeitschritte. Die farbig markierten zusammenhängenden Teilstücke der Trajektorie kennzeichnen das jeweils aktive Expertenmodul. Trotz der Einstreuung zufälliger Aktionen emergieren gegen Ende des Versuchs stabile quasizyklische Verhaltenssequenzen bei allen drei $\psi^{1,2,3}_{im}$.

feld erzwingt eine stetige Bewegung, weshalb die *treibenden Bereiche* um den Attraktor herum (zwangsläufig) häufiger besucht sind. ψ^3_{im} nutzt auch hier wieder die Dynamik des Vektorfelds aus und lässt sich hauptsächlich im Kreis treiben. Dabei führt es aber stetig Aktionen aus, welche dazu führen, dass der Radius der Kreisbewegung merklich größer wird.

Eine gleichmäßig Abdeckung des Zustandsraums kann bei beiden Verfahren nicht beobachtet werden. Dies ist auch nicht weiter verwunderlich, da im Modell, bis auf die Schnittstellen, keine Annahmen über die Struktur des Zustandsraums enthalten sind. Würde sich das Individuum zeitgleich ein Zustandsübergangsmodell mit aufbauen, so könnten beispielsweise durch Planung gezielt Bereiche angefahren werden, in denen der Vorhersagefehler noch hoch ist oder wo bisher nur wenige Experten ausgeprägt wurden. Die Suche im Zustandsraum ist in beiden Verfahren durch den Zufall getrieben, lediglich ψ_{im} bewertet rückwärtig gelungene Aktionen und führt diese wiederholt aus. Es gibt *keine explizite Motivation* nach unbesuchten Bereichen zu suchen; die bisherige Formulierung von Lernfortschritt reicht offensichtlich nicht aus, um dies als emergenten Effekt zu erzeugen. Allerdings führt ein zufällig entdeckter, unbesetzter Bereich zur Anlage eines neuen Experten, welcher wiederum durch seinen Lernfortschritt Belohnung produziert. Möglicherweise kann die gezielte Suche nach unbesuchten Bereichen unterstützt werden, indem eine extra Belohnung für das Anlegen neuer Experten vergeben wird. Die positiven (oder auch negativen) Auswirkungen dessen wurden im Zuge dieser Arbeit allerdings (noch) nicht untersucht.

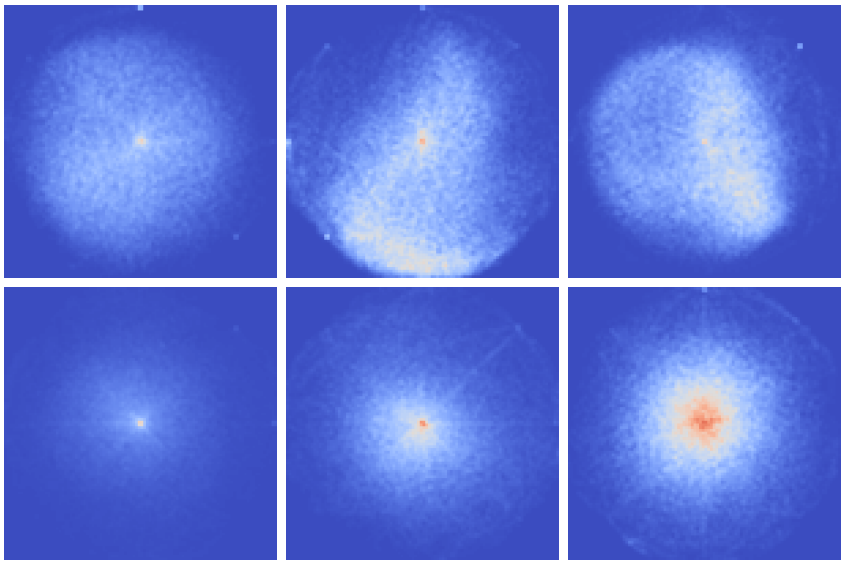

Abbildung 8.7: Häufigkeiten des Aufenthalts der sechs Individuen im jeweiligen motorischen Zustandsraum; Miniaturwelt 1:links, 2:mittig und 3:rechts. Intrinsische Motivation oben und zufällige Motivation unten. Blau: keine Anwesenheit, rot: häufige Anwesenheit.

Auswahl motorischer Aktionen

Die Aktionsauswahl wird unmittelbar durch die vergebene Belohnung beeinflusst. Demnach ist bei allen ψ_{rnd} die Aktionsauswahl erwartungsgemäß gleichverteilt. Allerdings ist je nach Morphologie die Intensität und Dauer der Ausübung unterschiedlich. In Abbildung 8.7 sind die motorischen Zustandsräume der Individuen abgebildet. Bei den Individuen ψ_{im} ist eine ungleichmäßige Verwendung des motorischen Raums zu beobachten. Mit Ausnahme der näherungsweise punktsymmetrischen Morphologie Nr. 1 sind bei den anderen Morphologien starke Präferenzen auszumachen. Beispielsweise hat ψ^2_{im} eine Präferenz für motorische Aktionen in Richtung 7 Uhr. Das hängt unmittelbar mit der Tatsache zusammen, dass für das Erreichen des zweiten Fixpunkts viel Kraft in diese Richtung aufgebracht werden muss, wohingegen das Zurückfallen fast kraftlos passiert (vgl. dazu nocheinmal Abbildung 8.5). In Abhängigkeit der Morphologie ergeben sich demnach Präferenzen für bestimmte Aktionen.

Weit tiefere Einsichten ergeben sich aus Abbildung 8.8. Dort ist die mittlere motorische Aktion, ebenfalls über die gesamte Dauer des Experiments gemittelt, aber in Abhängigkeit zum sensorischen Zustand abgetragen. Dabei sind zu allen drei Miniaturwelten die Individuen $\psi^{1,2,3}_{im}$ und $\psi^{1,2,3}_{rnd}$ in unmittelbarem Vergleich dargestellt. Es zeigt sich, dass der Algorithmus, sogar ohne die intrinsiche Motivation, den »Antrieb« erzeugt, stabile Situationen zu verlassen. Die motorischen Aktionen sind häufig gegen das Vektorfeld gerichtet (vgl. dazu die Vektorfelder der Morphologien in Abbildung 8.1). Dabei unterscheidet sich auch hier die Art der Ausübung motorischer Aktionen von $\psi^{1,2,3}_{rnd}$ und $\psi^{1,2,3}_{im}$ deutlich. Das Individuum ψ^1_{im} vollführt gegen Ende des Versuchs eine Oszillation, obwohl das anhand er Morphologie nicht zu erwarten wäre. Betrachtet man Welt Nr. 2, so erkennt man den leichtesten Pfad, den ψ^2_{im} gefunden hat, um

den Fixpunkt $\mathbf{x}_1^*$ zu verlassen. Das Individuum ψ_{im}^3 lässt sich im Kreis treiben und verstärkt mit seinen eigenen Aktionen die Oszillation und erhöht dabei Frequenz und Amplitude.

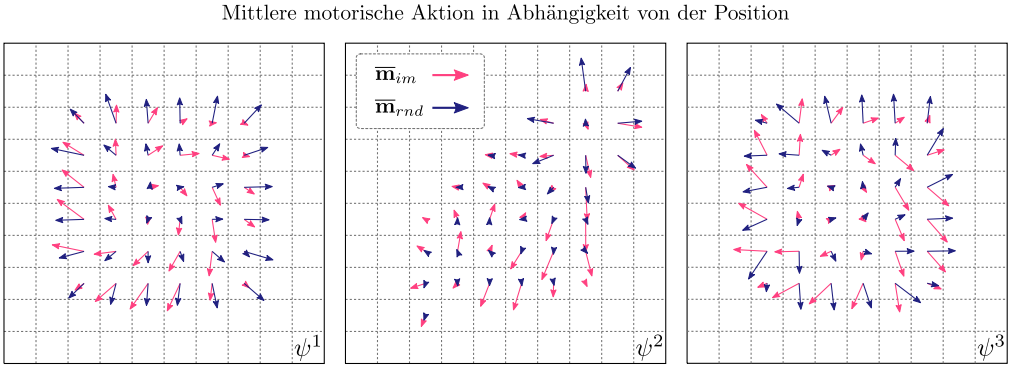

Abbildung 8.8: Gemittelte motorische Aktionen zum jeweiligen sensorischen Kontext.

Synapsenstruktur der Experten

In Abbildung 8.9 sind die Gewichte der FIR-Prädiktoren dargestellt. Die Abbildung links zeigt das Experten-Gas aus Experiment ψ_{im}^2, wobei die jeweiligen Experten gekennzeichnet sind. Die Dimension der zeitlichen Einbettung ist $K = 5$. In die Vorhersage gehen die Sensor- und Motorwerte (beide jeweils zweidimensional) und der Bias ein; die Vorhersage ist wie die Sensorwerte zweidimensional. Daher ergeben sich $2 \cdot (4 \cdot 5 + 1) = 42$ Gewichte pro Experte. In der Abbildung sind die Gewichte für die Sensordaten schwarz und dunkelblau; für die Motordaten mittelblau und hellblau und für den Bias rot gekennzeichnet. Die Synapsenstrukturen der verschiedenen Experten haben deutliche Unterschiede ausgeprägt. Auffällig ist, dass die Gewichte für die zweite Komponente der Voraussage, wenig Gebrauch von den Motorwerten macht. Das korrespondiert mit der Beobachtung, dass die motorischen Aktionen deutlich asymetrischer Natur sind.

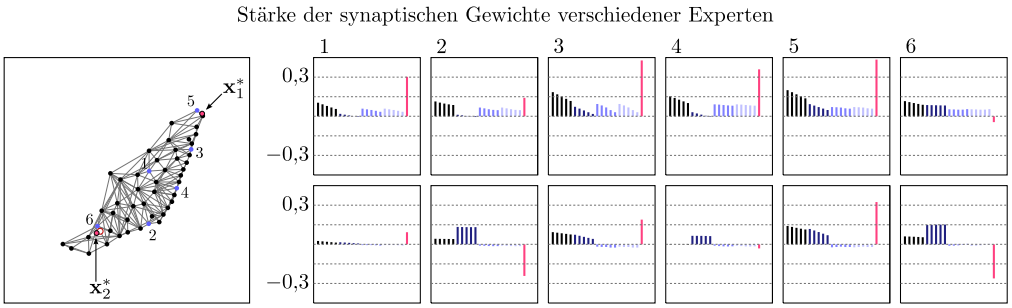

Abbildung 8.9: Dargestellt sind die Gewichte einzelner Expertenmodule aus Experiment ψ_{im}^2.

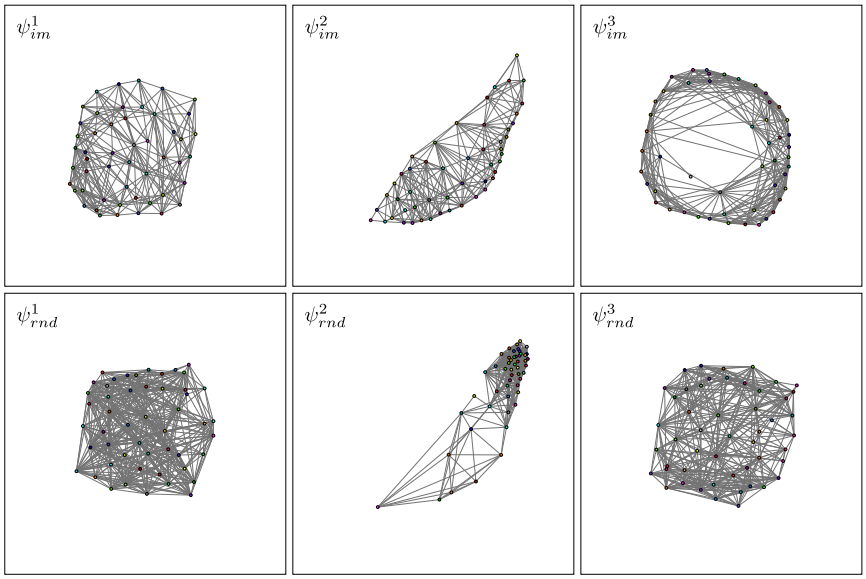

Abbildung 8.10: Multi-Experten-Struktur: Abgebildet sind die Zustände der Experten-Gase jeweils zum Ende des Versuchs. Die Graphen von ψ_{im} sind im Gegensatz zu denen von ψ_{rnd} dünner vernetzt. Anhand dicht liegender Experten können bevorzugte Pfade identifiziert werden.

Aufbau und Entwicklung der Multi-Experten-Struktur

Für die erste Morphologie bilden beide Individuen eine gleichmäßige Expertenstruktur aus. Dabei fällt allerdings auf, dass die Vernetzung von ψ^1_{im} wesentlich dünner ist. Bei jedem Zustandsübergang wird eine Kante angelegt, falls diese noch nicht existiert. Und diese werden auch nur nach und nach bei dem Experten mit der geringsten Nützlichkeit entfernt. Daraus lässt sich ableiten, dass bei ψ^1_{im} vermehrt intendierte Zustandsübergänge stattfinden, wohingegen ψ^1_{rnd} dicht vernetzt ist und das somit für viele zufällige Zustandsübergänge spricht.

Bei ψ^2_{rnd} ist auffällig, dass vermehrt Experten um den Fixpunkt $\mathbf{x}^*_1$ angelegt wurden. Die Experten-Dichte ist dabei äußerst inhomogen. Der Fixpunkt wird seltener verlassen, daher sind die Experten lokal sehr dicht, global aber ungleichmäßig. Dabei ist ein positiver Effekt des Expertengases zu bemerken: Obwohl der Fixpunkt vergleichsweise selten verlassen wird, bleiben die Experten außerhalb noch erhalten. Das Nützlichkeitsmaß wurde im Zuge der Modifikationen des GNG-U von der starken Zeitabhängigkeit freigestellt und somit können diese Zustände eine unbestimmte Weile überdauern. Die Experten von ψ^2_{im} verteilen sich hingegen über die leichtesten Verbindungsstrecken zwischen den Fixpunkten.

Bei den Individuen der dritten Morphologie ist auffällig, dass ψ^3_{im} nach einer Weile seine Experten fast ausnahmslos über den Bereich der treibenden Oszillation verteilt, wohingegen ψ^3_{rnd} den Raum gleichmäßig abdeckt. Während der Experimente fällt außerdem auf, dass von ψ^3_{im} neu angelegte Experten sich entweder rasch in den Kreis einordnen oder von der Nützlichkeitsabfrage eingeholt und wieder entfernt werden.

Auswertung des Belohnungssignals

Das Belohnungssignal wird aus den zuvor abgeleiteten Fehlersignalen der jeweiligen Gewinner-Experten zusammengesetzt. Die Verteilung des Belohnungssignals ist unsymmetrisch zur Null und hat jeweils halbseitig die Form einer Exponentialverteilung. Der relevante Wertebereich erstreckt sich näherungsweise über das Intervall $[-4 \cdot 10^{-4}, 8 \cdot 10^{-4}]$. Abbildung 8.11 zeigt einen typischen Ausschnitt des Belohnungssignals. Als Ersatz für das Belohnungssignal wurde dem Motivationssystem von ψ_{rnd}

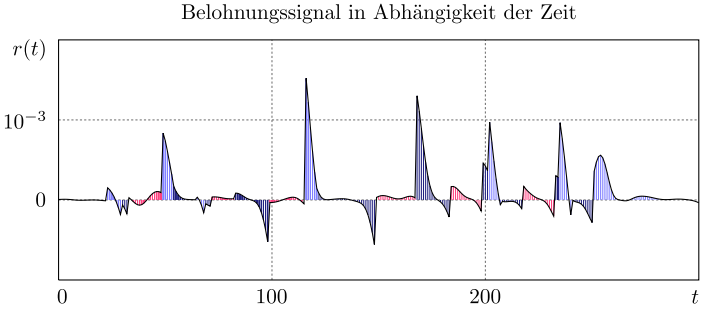

Abbildung 8.11: Erhaltene Belohnung in Abhängigkeit der Zeit. Die eingefärbten Bereiche markieren den jeweiligen Gewinner-Experten.

ein Rauschen übergeben. Da ein Rauschen keine kausale Information enthält, kann es zwangsläufig zu keinem intendierten Verhalten führen. Um die Ergebnisse von kausaler und zufälliger (nicht-kausaler) Belohnung vergleichbar zu halten wurde ein Zufallssignal produziert, was näherungsweise die gleiche Verteilung wie das echte Belohnungssignal hat. Um dazu die Verteilung zu approximieren, wurde ein gleichverteiltes Rauschen χ durch die Logarithmusfunktion, d. h. durch die Umkehrfunktion zur Exponentialfunktion, gegeben. Die beiden notwendigen Parameter $\lambda_{r,l}$ wurden dazu manuell ermittelt. Die Verteilung

$$\chi_R = \begin{cases} \log(-\chi)/\lambda_l & \chi < 0 \\ 0 & \chi = 0 \\ -\log(\chi)/\lambda_r & \chi > 0 \end{cases} \tag{8.7}$$

mit $\lambda_l = 5 \cdot 10^4$ und $\lambda_r = 10^4$ entspricht dann der als Rauschen angenäherten Verteilung des Belohnungssignals. Abbildung 8.12 zeigt beide Verteilungen im Vergleich. Im Allgemeinen konnten allerdings keine messbaren Unterschiede festgestellt werden, wenn anstatt der imitierten Verteilung der Einfachheit halber ein gleichverteiltes Rauschen im o. g. Intervall verwendet wurde.

Entwicklung des Vorhersagefehlers

Die Qualität der Vorhersage nimmt mit wachsender Anzahl der Experten zu. Um die Performanz in Bezug auf den Vorhersagefehler zu vergleichen wurde als Morphologie die Miniaturwelt Nr. 1 gewählt, da bei dieser aller anderen gemessenen Eigenschaften recht ähnlich waren. Die Abbildung 8.13 zeigt den durchschnittlichen Fehler der Systeme ψ^1 in Abhängigkeit zur maximalen Anzahl der Experten. Der mittlere quadratische

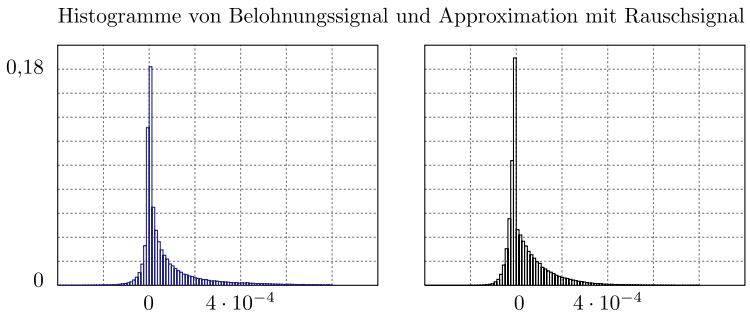

Abbildung 8.12: Vergleich der Histogramme von echtem Belohnungssignal und dessen Imitation, einem Rauschen mit ähnlicher Verteilung.

Fehler sinkt dabei schnell auf unter 10^{-6} und weiter ab. Allgemein kann festgestellt werden, dass ψ^1_{im} ab einer Anzahl von 10 Experten einen leicht geringeren Fehler hat und dieser über die Messreihen hinweg etwas weniger streut als bei ψ^1_{im}. Bei ψ_{im} sind allgemein temporär stark abfallende Fehlerraten zu beobachten, wenn durch exzessives Erproben ein und derselben Sequenz die Expertendichte auf diesem Pfad zunimmt. Der Fehler sinkt exponentiell, somit ließe sich gezielt die Aufmerksamkeit auf zu lernende Details lenken, wenn der Vorhersagefehler logarithmisch an das Belohnungsmodul übergeben werden würde. Insgesamt ergibt sich in Bezug auf den Vorhersagefehler der Eindruck, dass die Testfälle eher einfach für das System sind.

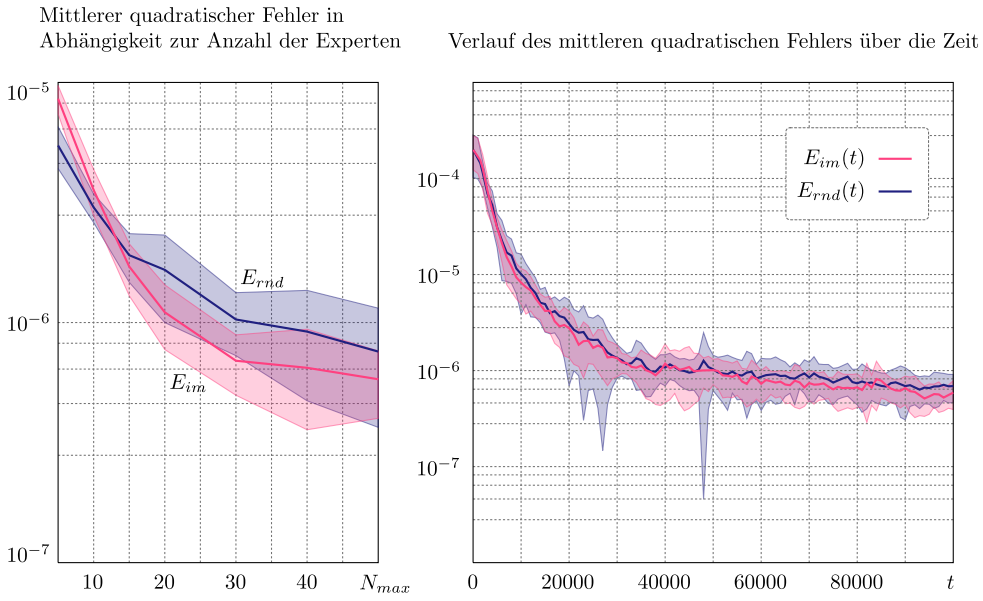

Abbildung 8.13: Mittlerer quadratischer Fehler in Abhängigkeit zur Anzahl der maximalen Experten N_{max}, gemessen jeweils nach 10^5 Zeitschritten (links). Entwicklung des mittleren quadratischen Fehlers in Abhängigkeit der Zeit, gemessen mit $N_{max} = 50$ Experten. Angabe von Mittelwert und Standardabweichung über jeweils 10 Messwerte.

Verhalten des Systems mit der Roboterplattform

Als Ergänzung zu den Resultaten aus den abstrakten Testszenarien konnten auf dem echten Roboter folgende Beobachtungen gemacht werden: Der Aufbau der Multi-Experten-Struktur verlief trotz erhöhter Dimensionen gut. Das System sollte vier Sensorwerte voraussagen und hatte dazu (8.5) und die Motorwerte des letzten Zeitschritts zur Verfügung. Bedingt durch anfänglich wenige Experten konnten zu Beginn des Versuchs überwiegend gleichbleibende Aktionen beobachtet werden. Die wenigen Experten führten erwartungsgemäß zu wenig Abwechslung. Je mehr Expertenmodule akquiriert wurden, desto komplexer wurden die beobachteten Bewegungen. Zwei markante Verhaltensweisen konnten öfter beobachtet werden: Zum einen eine Art Kopfstand, aus der Bauchlage hochgestämmt, mit teilweisem Überschlagen. Zum anderen ein mehrfaches Hin- und Herschaukeln, in der Rückenlage und mit weit abgestrecktem Bein, um Schwung zu holen. In späteren Lernstadien neigte das Verfahren dazu, sich längere Zeit an ein und derselben Stelle aufzuhalten, eine Art Lernfortschrittsnische. Es brauchte mitunter etwas, bis sich das Verfahren von selbst aus dieser Nische befreien konnte. Möglicherweis würde ein gezieltes Aufsuchen von Zuständen mittels Planung eine gleichmäßigeres Lernen ermöglichen.

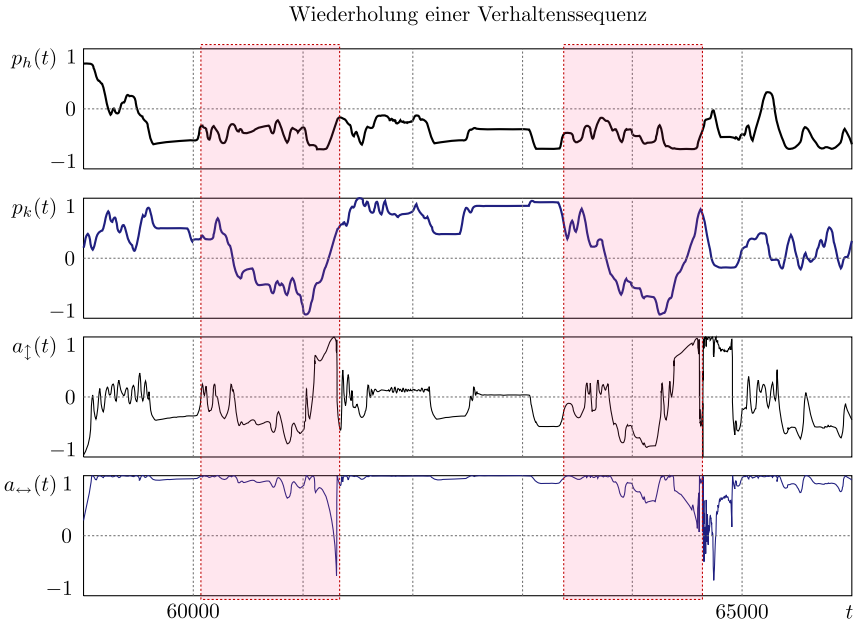

Abbildung 8.14: Spontane Emergenz von Verhaltenssequenzen: Abgebildet sind siebzig Sekunden des Experiments ψ^S_{im}. Eine Sequenz mit einer Länge von etwas mehr als zehn Sekunden erscheint etwa zwanzig Sekunden später erneut, wird dann aber durch eine Umkehrbewegung des Knie-Motors abgebrochen.

Für einen unabhängigen Beobachter ist das Verhalten von ψ^S_{im} (bisher) nur an kleinen unscheinbaren Details gegenüber ψ^S_{rnd} zu unterscheiden. Die spontan ergmergierenden Sequenzen waren im Gegensatz zu $\psi^{1,2,3}_{im}$ nicht stabil genug. Im besten Fall konnten bis zu 5 Wiederholungen erkannt werden (vgl. dazu Abbildung 8.14). Es konnte bislang

nicht eindeutig geklärt werden, welche Faktoren den stärksten Einfluss auf die Emergenz solcher Sequenzen hat. Dazu ist der Sprung von den abstrakten Miniaturwelten auf die vergleichsweise komplexe Hardware anscheinend zu groß. Für kommende Untersuchung erscheint es daher nötig, diese Lücke zu schließen und beispielsweise ein simuliertes oder echtes *physikalisches Pendel* oder einen Federschwinger als Testsystem zu untersuchen. Insgesamt ergaben sich quantitativ vergleichbare Ergebnisse auf der Roboterplattform, mit Außnahme der bei $\psi_{im}^{1,2,3}$ beobachteten stabilen Sequenzen.

Fazit

Ein wichtige Eigenschaft des Gesamtsystems, auch ohne intrinsiche Motivation, ist, dass es nicht zur Ruhe kommt. Der Antrieb ist eine inhärente Eigenschaft des Systems und es konnte, abgesehen von temporären Lernfortschrittsnischen, keine Stagnation des Verfahrens beobachtet werden. Stabile Zustände werden häufig verlassen, eine besondere Präferenz für instabile Lagen, d. h. balancierendes Verhalten, konnte dabei hingegen nicht beobachtet werden.

Die Individuen ψ_{rnd} finden viele Bereiche des sensorischen Zustandsraums, aber schwierige Pfade, welche aus stabilen Lagen herausführen, wie im Fall der Welt Nr. 2, entgehen ihm. Die Individuen ψ_{im} nutzen hingegen die Struktur des Zustandsraums gewissermaßen aus. Es ergeben sich emergente Verhaltenssequenzen wie ein Hin- und Herspringen zwischen zwei Fixpunkten oder eine Oszillation.

9 Zusammenfassung der Ergebnisse und Ausblick

Es wurde ein vollständiges Lernverfahren implementiert, welches aktiv und selbständig Wissen über Körper und Umwelt erwirbt. Vollständig in dem Sinne, dass stets das gesamte System und nicht nur ein bestimmtes Modul im Fokus stand. Das Individuum hat dabei nur das *Ziel*, beim Lernen selbst erfolgreich zu sein. Die Richtung wird nicht vorgegeben. Der Algorithmus ist an beschränkte Ressourcen anpassbar und auf unbegrenzte Laufzeit ausgelegt. Für die Implementation wurden großenteils neuronale und selbstregulierende Mechanismen verwendet. Die Fragestellung wurde dabei so gut es geht reduziert, ohne die Funktion des Gesamtsystems einzubüßen. Weiterhin wurde *versucht* die Annahmen über mögliche Morphologien so gering wie möglich zu halten; dabei mussten allerdings Abstriche gemacht werden: Im Vordergrund stand die ausschließliche Verwendung intrinsischer Motivation. Um die Funktionsfähigkeit des Gesamtsystems aufrecht zu erhalten, musste besonders die reale Plattform mit Schutzmechanismen, sowohl in Hard- als auch in Software, ausgestattet werden. Der Algorithmus ist wie sein zugrundeliegendes Rahmenwerk in [32] nach wie vor modular aufgebaut, sodass die Verbesserungen einzelner Module im Ganzen erprobt werden können. Im Folgenden werden die Erkenntnisse und Schlussfolgerungen zu den einzelnen Modulen, zum Gesamtsystems, sowie zu den Untersuchungen zusammengefasst.

Die Wahrnehmung und Zustandsidentifikation wird durch ein Multi-Experten-Netzwerk realisiert, wobei immer ein Experte den momentanen Zustand anzeigt. Die hier verwendete Struktur eines Experten ist ein einschichtiges neuronales Netz mit FIR-Synapsen, welches mittels Gradientenverfahren zur Laufzeit lernt. Die Struktur ist vergleichsweise einfach aber, bedingt durch die Verarbeitung der Signalhistorie, in der Lage dynamische Zustände voneinander zu trennen. Diese Struktur kann weiter ausgebaut werden. Dabei ist von besonderem Interesse, wie sich die Eigenschaften des Systems verändern, wenn man, statt vielen sehr einfachen, wenige komplex Experten verwendet.

Der Aufbau der Multi-Experten-Struktur wird als neuronales Gas implementiert. Diese Struktur ist auch unter begrenzten Ressourcen einsetzbar, weil sie flexibel und kontinuierlich das Wissen aktualisiert und unnütze Einheiten aussortiert werden, um wieder Platz für neues Wissen zu schaffen. Dazu wurde der GNG-U für die Verwendung zur Laufzeit generierter Sensorsignale angepasst und darüber hinaus eine adaptive Lernrate, im Sinne eines Lernkontingents, eingeführt. Zukünftig könnten die aufgebauten Kanten des Netzwerks als Zustandsübergänge modelliert werden, welche beispielsweise die mittlere motorische Aktivierung speichern, welche für den jeweiligen Zustandsübergang ausgeführt wurde. Damit ist eine Planung denkbar, welche das gezielte Anfahren bestimmter Zustände ermöglicht.

Durch die Identifikation des Experten mit dem geringsten Vorhersagefehler wird der

momentane (diskrete) Zustand des Systems festgelegt. Dabei bestanden anfangs Bedenken, ob der zeitliche Verlauf der Fehlersignale zusätzlich geglättet werden muss, um ein hektisches Hin- und Herschalten zwischen Zuständen zu vermeiden. Da das Minimum aller Vorhersagefehler zu jedem Zeitschritt neu bestimmt wird, besteht durchaus die Möglichkeit, dass ein eigentlich schlechterer Experte *zufällig* für einen Zeitschritt den geringsten Fehler hat. Nun könnte man annehmen, dass dies zu einer unbrauchbaren Spezialisierung der einzelnen Experten führt. Dem ist offenbar nicht so.

Durch die zeitliche Einbettung sind genügend Sensorwerte zur richtigen Unterscheidung verschiedener Zustände gegeben. Die Auswahl des Gewinner-Experten erfolgt ausschließlich anhand der Performanz zu jedem Zeitschritt. Trotz zufälliger *Einwürfe* eines scheinbar ungeeigneten Experten, welcher nur einen Zeitschritt lang Gewinner ist, entwickelt sich eine brauchbare Spezialisierung. Möglicherweise ist dies für die Spezialisierung sogar förderlich und macht diese robuster gegenüber Störungen. Diese Härte des Wettbewerbslernens konnte im Allgemeinen nicht als Problem identifiziert werden. Daher erscheint die Verwendung einer Hysterese beim Wechsel der Experten für ein erfolgreiche Spezialisierung nicht zwingend notwendig. Im Ergebnis wird dadurch eine präzisere Gesamtvorhersage der Sensorwerte erreicht, da kein Experte künstlich länger als Gewinner gilt, obwohl bereits ein bessere Vorhersage vorliegt – wenngleich dies auch nur für wenige Zeitschritte ist. Das Gesamtsystem ist damit auch reaktiver in Bezug auf die zu treffende Auswahl der nächsten motorischen Aktion. Eine Hysterese brächte zusätzliche zeitliche Verzögerungen und einzustellende Parameter.

Das System belohnt sich selbst für erfolgreichen Lernfortschritt. Belohnt wird dabei rein quantitativ das Absinken des Vorhersagefehlers. Dazu wurde der Vorhersagefehler durch ein speziell entworfenes Tiefpass-Ableitungs-Filter differenziert. Wichtig dabei ist, dass jedes kausale Filter, eine nicht vernachlässigbare Signalverzögerung erzeugt. Dieser Umstand wird bei der Verrechnung des Belohnungssignals beachtet. Um belohntes und nicht belohntes Verhalten zu vergleichen, wurde der Kontrollgruppe ψ_{rnd} ein nicht-kausales Rauschsignal mit einer angenäherten Verteilung präsentiert, also eine Art Placebo verabreicht. Weiterhin bleibt zu untersuchen, wie sich weitere gezielte Belohnungen auf andere qualitative Größen des Lernerfolgs auswirken. Beispielsweise könnte man das Anlegen eines neuen Experten belohnen oder, unter Voraussetzung eines Zustandsübergangsmodells, das Erreichen eines bisher wenig ausgeprägten Zustands belohnen.

Jeder Zustand trägt eine Liste der Bewertungen möglicher motorischer Aktionen mitsich. Um diese Bewertung aufzubauen und stetig zu aktualisieren wurde das Lernverfahren SARSA(λ) aus der Familie des bestärkenden Lernens gewählt. Für die Auswahl der Aktionen dient die Boltzmann-Selektion, welche mithilfe einer homöostatischen Lernregel die stochastische Auswahl in einen günstigeren Bereich regelt. Die ausgewählte motorische Aktion wird dann durch ein neuronales Feld in den kontinuierlichen Motorvektor überführt. Es muss aber bemerkt werden, dass der Ansatz der diskreten Aktionsauswahl grundlegend überdacht werden muss. Für die einfachen Modelle ist es ein adäquates Mittel und überschaubar zu implementieren, aber vor allem für die Analyse übersichtlich. Für viele motorische Freiheitsgrade muss hier aber eine kontinuierliche Alternative erprobt werden. Die Aktionswerte (Q-Matrix), hier als Tabelle implementiert, kann dabei unter Verwendung desselben Lernprinzips durch ein neuronales Netz ersetzt werden [38]. Das löst allerdings noch nicht das Problem, wie

aus einer diskreten Aktionsauswahl wieder kontinuierliche Motorwerte werden. Abhilfe gibt es hier durch das o. g. Zustandsübergangsmodell. Diskrete Aktionen könnten nun die von jedem Experten ausgehenden Kanten sein, an denen die mittlere motorische Aktion gespeichert ist. Eine Aktion auszuwählen hieße somit, sich für einen konkreten Zustandsübergang zu entscheiden.

Der vollständige Algorithmus ist echtzeitfähig. Dabei skaliert die Rechenzeit- und Speicherkomplexität linear mit der Anzahl der Expertenmodule. Die Auswahl des Gewinner-Experten wird durch hartes Wettbewerbslernen anhand des absoluten quadratischen Fehlers durchgeführt. Um die Rechenzeitkomplexität weiter zu reduzieren, kann die Auswahl des Gewinner-Experten durch einen gesonderten, selbständig lernenden Mechanismus erfolgen, einer Art *gating network* [21, 46]. Dieser könnte dynamisch die Anzahl konkurrierender Experten auf eine kleine Gruppe beschränken, sodass nicht alle Experten gleichzeitig *durchgerechnet* werden müssen. Ein solcher Mechanismus ließe sich über neuronale Felder, d. h. unter Hinzunahme einer Ortsangabe implementieren. Als eine solche dient möglicherweise der bereits in Abschnitt 5.1.4 separierte statische Anteil des Experten. Lokal liegende Experten konkurrieren und weiter entfernte werden inhibiert. Die inhibierten Experten würden dann zur Berechnung gar nicht erst hinzugezogen werden.

Der Algorithmus basiert auf unterschiedlichen Modulen, welche alle mit gewissenhaft einzustellenden Parametern ausgestattet sind. In der Summe kommt dabei eine beinahe unüberschaubare Menge an Stellschrauben auf und man ist dankbar für jeden Parameter, auf den verzichtet werden kann. Dabei hat es sich als nützlich erwiesen, die Einstellung einiger Parameter als Selbstregulation zu implementieren, wobei es schwierig ist, eine solche Lernregel wiederum ohne die Verwendung neuer Parameter zu entwerfen. Außer der Aussicht auf Belohnung durch erfolgreiches Lernen wurden dem Individuum keine weiteren Motivationen gegeben, um das Verhalten zu beeinflussen. Für die abstrakten Morphologien stellt das kein Problem dar, da diese relativ einfach waren und sich, durch Betrachtungen unter dem Zeitraffer, die ungünstigen Parametereinstellungen schnell identifizieren ließen. Auf der echten Hardware kann es dabei schnell passieren, dass ein Experiment nach einer wertvollen halben Stunde abgebrochen werden muss, weil sich das Lernverfahren aufgrund einer schlechten Parameterwahl in einem lokalen Minimum festfährt und augenscheinlich nicht in der Lage ist sich daraus *selbst zu befreien*. Daher ist es weitere Untersuchungen wert, um zu klären, wie für die Anwendung auf dem Roboter gezielt Mechanismen entworfen werden können, welche ein Festfahren vermeiden oder wieder lösen. Ein erster Versuch war es, den Aufenthalt in den diskreten Zuständen zu protokollieren und, bei längerer Anwesenheit in ein und demselben Zustand, die Auswahl der Aktionen Schritt für Schritt durch die Regulation der inversen Temperatur zu randomisieren.

Das Individuum beginnt den Lernprozess aus der Tabula-Rasa-Situation. Alle synaptischen Verbindungen werden zu Beginn zufällig initialisiert. Dabei ist zu beachten, dass auch zufällige Werte Informationen enthalten. Wenngleich auch diese Werte nicht aus dem Lernprozess selbst entstanden sind, bedeutet das, dass, von dieser Startinformation ausgehend, verschiedene Durchläufe desselben Experiments zu verschiedenen Endergebnissen führen können. Damit das System auf temporäre Lernerfolgsnischen reagieren kann, muss die Lernrate hoch eingestellt sein. Das erhöht sogleich die Reaktivität auf Rauschen. Daher müssen statistische Aussagen gegebenenfalls über mehrere

Experimente gemittelt werden. Für die Arbeit am Roboter bedeutet das einen nicht zu vernachlässigenden, erhöhten Aufwand.

Der vollständige Algorithmus wurde unter Variation verschiedener Morphologien erprobt. Die Untersuchungen an den einfachen zweidimensionalen abstrakten Morphologien zeigten bereits emergente Phänomene, die nicht zwingend zu erwarten gewesen sind. Dabei traten diese Phänomene sehr sensibel in Abhängigkeit der Parameter auf und waren meist erst in späteren Lernstadien zu beobachten.

Der sensorische Zustandsraum wurde im Hinblick auf den Aufenthalt des Individuums untersucht. Dabei stellte sich heraus, dass das intrinsisch motivierte Individuum gezielter in schwierige Bereiche vordringen konnte, welche durch zufällige Aktionen nur selten erreicht wurden. Dies ließe sich mit Sicherheit durch weitere gezielte Belohnungen verstärken. Mitnichten kann festgestellt werden, dass eine gleichmäßige Abdeckung des sensorischen Zustandsraums erreicht wurde. Weder das intrinsisch noch das zufällig motivierte Individuum konnte den sensorischen Raum vollständig erkunden. Das ist insofern auch nicht weiter verwunderlich, da explizit kein Modell über Art und Struktur des Raums zu Beginn vorhanden ist. Keines der Verfahren geht bei der Exploration systematisch vor. Beide sind durch den Zufall getrieben. Die bisherige Form der Quantifizierung des Lernfortschritts, die Ableitung des Fehlers, welche als Belohnung dient, kann keine systematische Suche erzeugen, d. h. dieser wünschenswerte Effekt emergiert nicht.

Der motorische Raum wurde erwartungsgemäß gleichmäßig vom zufällig belohnten Individuum ausgenutzt, wohingegen das intrinsisch motivierte Individuum gewissermaßen Präferenzen in Abhängigkeit der Morphologie entwickelte. Diese Präferenzen erwiesen sich als statistisch stabil über mehrere Experimente hinaus und bildeten eine Art motorischen Fingerabdruck der jeweiligen Morphologie.

Das Wachstum des neuronalen Gases zeigte sich bei den intrinsisch motivierten Individuen als insgesamt ausgewogener. Wenngleich auch diese Aussage bisher nicht durch ein quantitatives Maß belegt ist, kann alledem zum Trotz beobachtet werden, dass die Verteilung der Experten mehr auf die Dynamik der jeweiligen Morphologie zugeschnitten erscheint, als die Strukturen der zufällig motivierten Individuen. Diese neigen eher dazu, global homogene Vernetzungsstrukturen zu erreichen. Der Vorhersagefehler geht mit steigender Anzahl der Experten zurück. Bei den hier verwendeten Testszenarien sind bezüglich der intrinsischen Motivation bisher keine ausreichend reproduzierbaren Einflüsse auf den absoluten Vorhersagefehler auszumachen. Bei den intrinsisch motivierten Individuen sind temporär stark abfallende Fehlerraten zu beobachten, wenn spontane zeitweilig stabile Verhaltenssequenzen emergiert sind. Auch konnte örtlich lokal beobachtet werden, dass die stark nichtlinearen Randbereiche der abstrakten Morphologien weniger häufig besucht wurden, als es das zufällig motivierte Individuum tat. Dieser Effekt wird mit der gezielten Vermeidung von schwieriger erlernbaren Zusammenhängen erklärt.

Die Untersuchungen am Roboter sind im Gegensatz zu den abstrakten Morphologien wesentlich komplexer. Die Experimente müssen zudem in Echtzeit durchgeführt werden. Betrachtungen in Zeitlupe oder im Zeitraffer sind nur mit entsprechendem Aufwand realisierbar. Das erschwert die Erprobung der ohnehin schon komplizierten Parameterwahl. Außerdem sind mechanische Ausfälle enorm lästig. Die Roboterplattform Semni geht den richtigen Weg in Richtung einer Reduzierung der Komplexi-

tät. Vor allem in Bezug auf die Vermeidung von selbst zugeführten Schäden kann der Experimentator sich gelassen auf die eigentlichen Experimente konzentrieren und wird von aufwändigen Vorsichtsmaßnahmen freigestellt. Nichtsdestotrotz gestaltet sich die Durchführung von Langzeitexperimenten, *ohne stetigen Betreuungsaufwand*, als schwierig. Der Roboter muss ggf. aus einer Verkantung befreit werden oder verheddert sich im Kabel. Wünschenswert wären weitere Möglichkeiten die Experimentieranordnung zu stabilisieren, um die Ergebnisse reproduzierbarer zu machen und den Betreuungsaufwand weiter zu reduzieren. Die Roboterplattform selbst zeigte sich erfreulich robust und bis auf ein eingequetschtes Kabel konnten glücklicherweise keinerlei Schäden beklagt werden.

Ein weiterer überlegenswerter Ansatz wäre es, die Frage zu stellen, inwieweit die Experimentatorethik bei dieser Art Untersuchungen ausgesetzt werden kann. Gedacht der Fall der Roboter Semni wäre mit entsprechender Sensorik und den Mechanismen ausgestattet, die ihm ermöglichen Berührungen zu detektieren. Damit könnte man einen Mechanismus implementieren, welcher die motorischen Aktionen zeitweise aussetzt, wenn der Experimentator eingreift. Somit kann, ohne die Gefahr von Quetschungen, gezielt in den Ablauf des Experiments eingegriffen werden und der Roboter angeleitet, oder aus verzwickten Situationen befreit werden. Hilfestellungen von Außen sind für aktiv lernende, biologische Individuen nicht ungewöhnlich. In das Experiment eingreifen? Daran ist im Wesentlichen nichts verwerflich. Imitation ist für das Lernen bei vielen Spezies essentiell und Lernen von anderen führt in der Natur bekanntlich zu schnellem Lernerfolg.

Es konnte gezeigt werden, dass die intrinsische Motivation, über eine zufällige Exploration hinaus, in der Lage ist den Vorhersagefehler zu senken, motorische Präferenzen herauszuarbeiten und ermergente Verhaltenssequenzen zu erzeugen. Somit ist dieses Verfahren für die Exploration des Zustandsraums einsetzbar und kann mit dem Aufbau eines Zustandsübergangsmodells kombiniert werden. Die intrinsische Motivation bleibt aber ausbaufähig. Die konkrete Formulierung als Ableitung des Vorhersagefehlers kann mit Sicherheit noch verallgemeinert werden; beispielsweise auf alle Module des Systems, indem alles was außerdem noch als Lernfortschritt gewertet werden kann, zusätzlich belohnt wird.

Wenngleich das System nachweislich etwas lernt, d. h. die Multi-Experten-Struktur sich aufbaut und der Vorhersagefehler sinkt, bleiben durch das Fehlen von Vergleichsmöglichkeiten die Prozesse oft schwer durchschaubar. Der Lernvorgang des Systems entzieht sich weitestgehend der Anschauung. Der Lernende ist sich vollständig selbst überlassen, insofern ist es fraglich, inwieweit ein Lernfortschritt, abgekoppelt von extrinsischen Motiven und unter vollständigem Ausschluss von Interaktion, überhaupt vom unabhängigen Beobachter interpretiert werden kann. Wie sollte das Verhalten im Idealfall aussehen, wonach man sucht?

A Anhang

A.1 Mathematische Ergänzungen

A.1.1 Sigmoide Ausgangsfunktion

Oft findet man in der Literatur andere Ausgangsfunktionen, wie beispielsweise die Sigmoidfunktion (im Englischen auch oft *squashing function* oder *logistic function*). Ist die Verwendung dieser alternativen Ausgangsfunktion erforderlich, ist es gegebenenfalls nützlich die bestehenden Mittel wiederzuverwenden, beispielsweise wenn bereits der Tangens Hyperbolicus effizient durch eine stückweise lineare Approximation implementiert wurde. Die Sigmoidfunktion ist definiert als

$$\phi(x) = \frac{1}{1 + e^{-x}} \tag{A.1}$$

und lässt sich mithilfe des Tangens Hyperbolicus als

$$\phi(x) = \frac{\tanh\left(\frac{1}{2}x\right) + 1}{2} \tag{A.2}$$

ausdrücken.

Die Darstellung (A.1) der Sigmoidfunktion unter Verwendung der Exponentialfunktion kann wie folgt in die Darstellung (A.2) umgeformt werden:

$$\begin{aligned}
\phi(x) =& \frac{1}{1 + e^{-x}} = \frac{e^{\frac{1}{2}x}}{e^{\frac{1}{2}x} + e^{-\frac{1}{2}x}} = \frac{e^{\frac{1}{2}x}}{e^{\frac{1}{2}x} + e^{-\frac{1}{2}x}} - \frac{1}{2} + \frac{1}{2} \\
=& \frac{e^{\frac{1}{2}x}}{e^{\frac{1}{2}x} + e^{-\frac{1}{2}x}} - \frac{1}{2}\left(\frac{e^{\frac{1}{2}x} + e^{-\frac{1}{2}x}}{e^{\frac{1}{2}x} + e^{-\frac{1}{2}x}}\right) + \frac{1}{2} \\
=& \frac{e^{\frac{1}{2}x} - \frac{1}{2}e^{\frac{1}{2}x} - \frac{1}{2}e^{-\frac{1}{2}x}}{e^{\frac{1}{2}x} + e^{-\frac{1}{2}x}} + \frac{1}{2} \\
=& \frac{1}{2}\left(\frac{e^{\frac{1}{2}x} - e^{-\frac{1}{2}x}}{e^{\frac{1}{2}x} + e^{-\frac{1}{2}x}}\right) + \frac{1}{2} \\
=& \frac{\tanh\left(\frac{1}{2}\mathrm{x}\right) + 1}{2}
\end{aligned} \tag{A.2}$$

Kontinuierliches Neuronenmodell

Häufig begegnet man in der Literatur einem zeitkontinuierlichen Neuronen-Modell. Ein einzelnes Neuron j wird dann in Form einer Differentialgleichung wie

$$\tau \dot{y}_j = -y_j + I_j + b_j + \sum_{i=1}^{N} w_{ji}\phi(y_i) \tag{A.3}$$

mit der sigmoiden Funktion ϕ, der externen Netzeingabe I und dem Bias b dargestellt. Der Vorteil einer kontinuierlichen Darstellung liegt in theoretisch beliebig feinen Zustandsübergängen. Allerdings erfordert die Verwendung kontinuierlicher Modelle in der Praxis eine numerische Behandlung beispielsweise mit dem expliziten Euler-Verfahren und die Angabe der Zeitkonstanten τ, welche die Geschwindigkeit der zeitlichen Entwicklung des Modells angibt. Die Übersetzung in ein zeitdiskretes Neuronenmodell erfolgt durch Festlegung von τ und das Ersetzen des Differentialquotienten $\dot{y} = \mathrm{d}y/\mathrm{d}t$ durch $y(t+\tau) - y(t)$ [5].

A.1.2 Herleitung der Infomax-Lernregel

Die erste Hälfte der Herleitung ist analog zu [2, 3], wohingegen der Rest sich auf die Ausgangsfunktion *Tangens Hyperbolicus* bezieht. Ursprünglich wurde die Lernregel für die Standardsigmoide angefertigt. Die Herleitung hier bezieht sich auf das Neuronenmodell (3.3) mit *einem* Eingangsgewicht und dem Bias. Für weitere Architekturen finden sich in der o. g. Literatur analoge Herleitungen.

Betrachtet man den Ein- und Ausgang eines Neurons als Zufallsgrößen X und Y, so ergibt sich die Stärke des statistischen Zusammenhangs von X und Y als die sogenannte Transinformation. Diese ist definiert als

$$I(Y;X) = H(Y) - H(Y|X),$$

wobei $H(Y)$ die Ausgangsentropie (Empfangs-Entropie) und $H(Y|X)$ die Fehlinformation ist. Möchte man nun die Transinformation mithilfe eines Gradientenverfahrens maximieren, so leitet man dazu partiell nach dem Gewicht w ab:

$$\frac{\partial I(Y;X)}{\partial w} = \frac{\partial H(Y)}{\partial w}$$

Das Modell des Neurons ist deterministisch, daher fällt die Fehlinformation aus der Berechnung heraus. Jedwedes Rauschen ist bereits im Eingangssignal enthalten und lässt sich hier nicht getrennt betrachten. Daher wird die Transinformation maximiert, wenn die Ausgangsentropie maximiert wird. Die Ausgangsentropie ist definiert als

$$H(y) = -\,\mathrm{E}\left[\ln f_y(y)\right],$$

wobei $f_y(y)$ die Wahrscheinlichkeitsdichte des Ausgangs ist und $E[\cdot]$ den Erwartungswert kennzeichnet. Mit Hilfe der Beziehung

$$f_y(y) = \frac{f_x(x)}{|\partial y/\partial x|}$$

führt das zu der Umformung

$$H(y) = -\,\mathrm{E}\left[\ln f_y(y)\right] = \mathrm{E}\left[\ln\left|\frac{\partial y}{\partial x}\right|\right] - \mathrm{E}\left[\ln f_x(x)\right].$$

Der letzte Term ist nichts weiter als die Eingangsentropie, sie fällt bei einer Ableitung nach dem Gewicht aus der Betrachtung heraus, da sie nicht mit einer Änderung von w beeinflusst werden kann. Die Änderung des Gewichts

$$\Delta w \propto \frac{\partial H}{\partial w} = \frac{\partial}{\partial w}\left(\ln\left|\frac{\partial y}{\partial x}\right|\right) = \left(\frac{\partial y}{\partial x}\right)^{-1}\frac{\partial}{\partial w}\left(\frac{\partial y}{\partial x}\right)$$

ist proportional zur Ableitung der Entropie nach dem Gewicht und kann mit Hilfe der Logarithmusgesetze vereinfacht werden. Die Ausgangsfunktion ist der Tangens Hyperbolicus, das Modell für das Neuron

$$y = \tanh(wx + b)$$

und mit den partiellen Ableitungen

$$\frac{\partial y}{\partial x} = w(1+y)(1-y) = w\left(1-y^2\right)$$

sowie

$$\begin{aligned}
\frac{\partial}{\partial w}\left(\frac{\partial y}{\partial x}\right) &= \frac{\partial}{\partial w}\left[w\left(1-y^2\right)\right] \\
&= \left(1-y^2\right) + w\frac{\partial}{\partial w}\left(1-y^2\right) \\
&= \left(1-y^2\right) - w\frac{\partial}{\partial w}y^2 \\
&= \left(1-y^2\right) + 2wxy\left(1-y^2\right)
\end{aligned}$$

ergibt sich somit:

$$\begin{aligned}
\Delta w \propto & \left(\frac{\partial y}{\partial x}\right)^{-1}\frac{\partial}{\partial w}\left(\frac{\partial y}{\partial x}\right) = \frac{\left(1-y^2\right)+2wxy\left(1-y^2\right)}{w\left(1-y^2\right)} \\
\Delta w \propto & \frac{1}{w} - 2xy
\end{aligned}$$

Analog dazu verfährt man für das Bias-Gewicht b:

$$\begin{aligned}
\frac{\partial}{\partial b}\left(\frac{\partial y}{\partial x}\right) &= \frac{\partial}{\partial b}\left[w\left(1-y^2\right)\right] \\
&= w\frac{\partial}{\partial b}\left(1-y^2\right) \\
&= -w\frac{\partial}{\partial b}y^2 \\
&= -2wy\left(1-y^2\right)
\end{aligned}$$

$$\Delta b \propto \left(\frac{\partial y}{\partial x}\right)^{-1}\frac{\partial}{\partial b}\left(\frac{\partial y}{\partial x}\right) = \frac{-2wy\left(1-y^2\right)}{w\left(1-y^2\right)} = -2y$$

Abschließend ergeben sie die beiden Lernregeln für w und b:

$$\begin{aligned}
\Delta w &:= \eta_w\left(\frac{1}{w} - 2xy\right) \\
w(t+1) &:= w(t) + \Delta w(t)
\end{aligned}$$

$$\begin{aligned}
\Delta b &:= \eta_b\left(-y\right) \\
b(t+1) &:= b(t) + \Delta b(t)
\end{aligned}$$

A.2 Filterkoeffizienten

In Kapitel 6 wurden zwei Ableitungsfilter mit Tiefpasseigenschaften hergeleitet. Für die auf $N = 21$ Koeffizienten optimierten Varianten sind hier der Vollständigkeit halber die Koeffizienten aufgelistet.

Hamming	Selesnick
0,0015855	0,0021235
0,0026451	0,0061818
0,0051297	0,0096908
0,0090674	0,0123388
0,0137934	0,0138904
0,0180857	0,0142077
0,0205241	0,0132627
0,0199573	0,0111392
0,0159190	0,0080260
0,0088435	0,0041995
0,0000000	0,0000000
−0,0088435	−0,0041995
−0,0159190	−0,0080260
−0,0199573	−0,0111392
−0,0205241	−0,0132627
−0,0180857	−0,0142077
−0,0137934	−0,0138904
−0,0090674	−0,0123388
−0,0051297	−0,0096908
−0,0026451	−0,0061818
−0,0015855	−0,0021235

Tabelle A.1: Die Filterkoeffizienten (6.9) (Hamming), (6.8) (Selesnick).

A.3 Biologische Plausibilität

Um den Mechanismen für die Verhaltensweisen biologischer Individuen auf den Grund zu gehen, werden vielerorts unterschiedliche Methoden verwendet. So ist es die Aufgabe der Psychologie durch Beobachtungen von gezielten Verhaltensexperimenten und durch Empirie Nachweise für ihre Thesen zu erbringen. Der Anspruch dabei ist es, möglichst minimalinvasive Methoden zu finden, um den natürlichen Ablauf nicht zu durch die eigentliche Experimentieranordnung zu verfälschen. Einen völlig anderen Ansatz verfolgen die medizinischen Wissenschaften. Bei ihren Untersuchungen gehen sie einen reduktionistischen[1] Weg. Auch sie gehen vom bestehenden Individuum aus, untersuchen dabei aber die vorhandenen Strukturen bis ins kleinste Detail, um aus der

[1] Das Adjektiv *reduktionistisch* ist hierbei als relativ zur Psychologie zu betrachten. Die Neurowissenschaften und Medizin im Allgemeinen sehen sich meist mit emergenten Phänomen konfrontiert. Die eigentlichen Reduktionisten sind jedoch (noch) die Physiker.

Funktion des Einzelteils die Funktion des großen Ganzen abzuleiten. Eine dritte Herangehensweise vertreten die Kognitionswissenschaften. Ihr Anspruch ist es, die Prozesse des Zentralnervensystems (ZNS) durch den *Nachbau* desselbigen zu verstehen. Dies geschieht unter Einsatz teils massiver Rechenleistung digitaler Computer.

Nicht selten kommt es dabei vor, dass gefundene Algorithmen als »biologisch unplausibel« gelten [28]. Die Begründung läuft meist auf die Aussage hinaus, das bis dato keine biologischen Strukturen, z. B. im menschlichen Gehirn, gefunden wurden, welche exakt die *Berechnungen* durchführen, wie sie der Algorithmus macht. Die Eigenschaft *biologische Plausibilität* hat sich für die Rechtfertigung von Aufbau, Vergleich oder Ausschluss unterschiedlicher Lernverfahren gemausert, doch bleibt sie weiterhin eine »heiße Kartoffel« und immer an der Grenze missverstanden zu werden. Wenn eine Lernregel oder ein ganzer Algorithmus als biologisch plausibel gilt, sollte das in eigentlicher Absicht nur heißen, dass das Ergebnis, z. B. ein bestimmtes Verhalten, auch aus der Biologie bekannt ist oder schon beobachtet und untersucht wurde, und dass somit der Algorithmus als ein Modell für dieses Verhalten biologisch plausibel ist. Zur Verwechslung neigt aber die Auffassung, dass auch der Algorithmus, d. h. die Rechenvorschrift selbst, in der Art biologisch plausibel sein solle, als dass bereits aus den medizinischen Wissenschaften diese Art der Berechnung in biologischen informationsverarbeitenden Systemen nachgewiesen wurde. Somit stütze man sich aber auf die Ergebnisse einer anderen (evtl. basaleren) Disziplin und blockiere unnötigerweise den gedanklichen Fortschritt der Eigenen.

Die Auffassung ist aber auch insofern trügerisch, als das Neuronen als Einzelteil einer universellen *Rechenmaschine* im Stande sind vielfältigste Funktionen zu übernehmen und diese oft auch nur unter gewissen Schwierigkeiten identifiziert werden können. Ganz davon abgesehen, dass die Evolution viele Wege der konkreten Ausgestaltung findet, welche näherungsweise dasselbe Ergebnis haben können. Beispielsweise lassen sich aus Neuronen Filter bauen, die der klassischen Filtertheorie in nichts Wesentlichem nachstehen. Es lassen sich Zustandsautomaten und Boolsche Logik abbilden oder Funktionsapproximationen und Zeitreihenvorhersage durchführen, um nur einige Beispiele zu nennen. Insofern kann nur selten mit Sicherheit gesagt werden, das dieses oder jenes Verfahren biologisch nicht plausibel ist. Daher gebietet es sich mit einer gewissen Vorsicht dieses Gütesiegel zu vergeben und biologisch scheinbar unplausible Lernregeln nicht vorschnell auszuschließen.

Als unglückliches populäres Beispiel sei hier exemplarisch die in Abschnitt 3.2.2 erklärte *Backpropagation*-Lernregel benannt. Ihr wird die Nichtlokalität der für das Lernen verwendeten Informationen zur Last gelegt, da *bisher* nicht hinreichend klar ist, ob und wie das ZNS globale Informationen dieser Art verarbeitet. Weiterhin sei nicht klar wie vom ZNS eine Fehlerrückführung über mehrere Neuronen hinweg vollzogen werden solle. Nichtsdestoweniger ist nicht auszuschließen, dass ein allein auf lokalem Informationsaustausch beruhendes emergentes Lernverfahren bereits näherungsweise dies im ZNS leistet und das Gradientenverfahren eine Näherung (oder gar eine Verallgemeinerung) davon ist. Die Diskussionen ob biologisch plausibel oder nicht verlaufen daher, so auch diese hier, äußerst spekulativ, sogar teils unwissenschaftlich, und sollten weitestgehend vermieden werden. Alledem zum Trotz ist in dieser Arbeit ein merklich überproportionaler Anteil biologisch inspirierter Lernregeln und Paradigmen vertreten, was jedoch mehr dem Geschmack des Autors geschuldet ist.

Literaturverzeichnis

[1] Back, Andrew D. ; Tsoi, Ah C.: FIR and IIR Synapses, A New Neural Network Architecture for Time Series Modelling. In: *Neural Computation* 3 (1991), Nr. 3, S. 375 – 385

[2] Bell, Anthony J. ; Sejnowski, Terrence J.: A Non-linear Information Maximisation Algorithm that Performs Blind Separation. In: *Advances in Neural Information Processing Systems 7*, MIT Press, 1995, S. 467 – 474

[3] Bell, Anthony J. ; Sejnowski, Terrence J.: An Information-Maximization Approach to Blind Separation and Blind Deconvolution. In: *Neural Computation* 7 (1995), S. 1129 – 1159

[4] Churchland, Patricia S. ; Sejnowski, Terrence J.: *Grundlagen zur Neuroinformatik und Neurobiologie* (The Computational Brain). Vieweg, 1997

[5] Doya, Kenji: Bifurcation of Recurrent Neural Networks in Gradient Descent Learning. In: *IEEE Transactions on Neural Networks* (1993)

[6] Doya, Kenji: Universality of Fully-Connected Recurrent Neural Networks / IEEE Transactions on Neural. 1993. – Forschungsbericht

[7] Doya, Kenji: Metalearning and Neuromodulation. In: *Neural Networks* 15 (2002)

[8] Doya, Kenji: Recurrent Networks: Learning Algorithms. In: *The Handbook of Brain Theory and Neural Networks*. 2. MIT Press, 2002

[9] Dörner, Dietrich: *Bauplan für eine Seele*. Rowohlt, 2001. – ISBN 3–499–61193–7

[10] Elman, Jeffrey L.: Finding Structure in Time. In: *Cognitive Science* 14 (1990), Nr. 2, S. 179 – 211

[11] Fiori, Simone: Hybrid independent component analysis by adaptive LUT activation function neurons. In: *Neural Networks* 15 (2002), Nr. 1, S. 85–94

[12] Fritzke, Bernd: A Growing Neural Gas Network Learns Topologies. In: *Advances in Neural Information Processing Systems 7*, MIT Press, 1995, S. 625 – 632

[13] Fritzke, Bernd: A Self-Organizing Network That Can Follow Non-Stationary Distributions. In: *Proc. of ICANN-97, International Conference on Artificial Neural Networks*, Springer, 1997, S. 613 – 618

[14] Fritzke, Bernd: *Vektorbasierte Neuronale Netze*. Erlangen, Friedrich-Alexander-Universität Erlangen-Nürnberg, Habilitationsschrift, 1998

[15] Gollin, Michael: *Implementation einer Bibliothek für Reinforcement Learning und Anwendung in der RoboCup-Simulationsliga.* : Institut für Informatik, Humboldt-Universität zu Berlin, 2005. – Diplomarbeit

[16] Hamming, Richard W.: *Digital Filters.* 3rd. Prentice Hall, 1989. – Paperback reprint: Courier Dover Publications, 1998

[17] Hild, Manfred: *Neurodynamische Module zur Bewegungsteuerung autonomer mobiler Roboter*, Institut für Informatik, Humboldt-Universität zu Berlin, Diss., 2007

[18] Hild, Manfred ; Meissner, Robin ; Spranger, Michael: Humanoid Team Humboldt, Team Description / Humboldt-Universität zu Berlin, Institut für Informatik, LFG Künstliche Intelligenz. 2007. – Forschungsbericht

[19] Hoffmann, Norbert: *Kleines Handbuch: Neuronale Netze.* Vieweg, 1993

[20] Höfer, Sebastian ; Hild, Manfred: Using Slow Feature Analysis to Improve the Reactivity of a Humanoid Robot'S Sensorimotor Gait Pattern. In: *International Conference on Neural Computation* (2010)

[21] Jacobs, Robert A. ; Jordan, Michael I. ; Nowlan, Steven ; Hinton, Geoffrey E.: Adaptive Mixtures of Local Experts. In: *Neural Computation* 3 (1991), S. 79 – 87

[22] Jaeger, Herbert ; Haas, Harald: Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication. In: *Science* 304 (2004), Nr. 5667, S. 78 – 80

[23] Kaplan, Frédéric ; Oudeyer, Pierre-Yves: In Search of the Neural Circuits of Intrinsic Motivation. In: *Frontiers in Neuroscience* (2007)

[24] Kubisch, Matthias: *Modellierung und Simulation nichtlinearer Motoreigenschaften.* : Institut für Informatik, Humboldt-Universität zu Berlin, 2008. – Studienarbeit

[25] Lapedes, Alan S. ; Farber, Robert M.: How Neural Nets Work. In: *NIPS*, 1987, S. 442 – 456

[26] Laughlin, Robert B.: *Abschied von der Weltformel: Die Neuerfindung der Physik.* 4. Piper, 2007

[27] Martius, Georg ; Fiedler, Katja ; Herrmann, J. M.: Structure from Behavior in Autonomous Agents. In: *IEEE International Conference on Intelligent Robots and Systems*, 2008, S. 858 – 862

[28] Mazzoni, Pietro ; Anderson, Richard A. ; Jordan, Michael I.: A More Biologically Plausible Learning Rule Than Backpropagation Applied to a Network Model of Cortical Area 7a. In: *Cerebral Cortex* 1 (1991), S. 293 – 307

[29] Moreira, M. ; Fiesler, E.: Neural Networks with Adaptive Learning Rate and Momentum Terms / Institut Dalle Molle D'intelligence artificielle perceptive (IDIAP). 1995. – Forschungsbericht

[30] Nishide, Shun ; Ogata, Tetsuya ; Yokoya, Ryunosuke ; Tani, Jun ; Komatani, Kazunori ; Okuno, Hiroshi G.: Active sensing based dynamical object feature extraction. In: *IROS*, 2008, S. 1–7

[31] Oudeyer, Pierre-Yves ; Kaplan, Frédéric: How Can We Define Intrinsic Motivation? In: *International Conference on Epigenetic Robotics*, 2008

[32] Oudeyer, Pierre-Yves ; Kaplan, Frédéric ; Hafner, Verena V.: Intrinsic Motivation Systems for Autonomous Mental Development. In: *IEEE Transactions on Evolutionary Computation* 11 (2007)

[33] Pasemann, Frank ; Hild, Manfred ; Zahedi, Keyan: SO(2)-Networks as Neural Oscillators. In: *Proc. of Int. Work-Conf. on Artificial and Natural Neural Networks (IWANN)* (2003), S. 144 – 151

[34] Pearlmutter, Barak A.: Gradient Calculations for Dynamic Recurrent Neural Networks: A Survey. In: *IEEE Transactions on Neural Networks* 6 (1995), S. 1212 – 1228

[35] Pfeifer, Rolf ; Iida, Fumiya: Embodied Artificial Intelligence: Trends and Challenges. In: *Embodied Artificial Intelligence*, 2003, S. 1–26

[36] Principe, Jose C. ; Vries, Bert de ; Oliveira, Pedro G.: The Gamma Filter–A New Class of Adaptive IIR Filters with Restricted Feedback. In: *IEEE Transactions on Signal Processing* 41 (1993)

[37] Rumelhart, David E. ; Hinton, Geoffrey E. ; Williams, Ronald J.: Learning Internal Representations by Error Propagation. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* Bd. 1: Foundations. MIT Press, 1986, S. 318 – 362

[38] Rummery, G. A. ; Niranjan, M.: On-Line Q-Learning Using Connectonist Systems / CUED/F-INFENG/TR 166, Cambridge University Engineering Department, England. 1994. – Forschungsbericht

[39] Schiffmann, Wolfram ; Joost, Merten ; Werner, Randolf: Comparison of Optimized Backpropagation Algorithms. In: *Proc. of ESANN'93, Brussels*, 1993, S. 97–104

[40] Selesnick, Ivan W.: Narrowband Lowpass Digital Differentiator Design. In: *Electrical and Computer Engineering, Polytechnic University* (2006)

[41] Shannon, Claude E.: A Mathematical Theory of Communication. In: *The Bell System Technical Journal* 27 (1948), S. 379–423,623–656

[42] Steil, Jochen J.: Backpropagation-Decorrelation: Online Recurrent Learning With O(N) Complexity. In: *Proc. IJCNN* Bd. 1, 2004, S. 843 – 848

[43] Steil, Jochen J.: Online Reservoir Adaptation by Intrinsic Plasticity for Backpropagation-Decorrelation and Echo State Learning. In: *Neural Networks* 20 (2007), Nr. 3, S. 353 – 364

[44] Stephan, André: *Visualisierung und Neuronales-Gas-Lernen von Sensordaten des humanoiden A-Serie Roboters.* : Institut für Informatik, Humboldt-Universität zu Berlin, 2010. – Diplomarbeit

[45] Sutton, Richard S. ; Barto, Andrew G.: *Reinforcement Learning: An Introduction.* MIT Press, 1998

[46] Toussaint, Marc: A Neural Model for Multi-Expert Architectures. In: *Proc. of the Int. Joint Conference on Neural Networks (IJCNN)* Bd. 3, 2002, S. 2755 – 2760

[47] Toussaint, Marc: A Sensorimotor Map: Modulating Lateral Interactions for Anticipation and Planning. In: *Neural Computation* 18 (2006), S. 1132 – 1155

[48] Tsung, Fu-Sheng ; Cottrell, Garrison W.: Phase Space Learning. In: *Neural Information Processing Systems*, 1994

[49] Turrigiano, Gina G. ; Nelson, Sacha B.: Homeostatic Plasticity in the Developing Nervous System. In: *Nature Reviews, Neuroscience* (2004)

[50] Werbos, Paul J.: Backpropagation Through Time: What It Does and How to Do It. In: *Proceedings of the IEEE* 78 (1990), Nr. 10, S. 1550 – 1560

[51] Wikipedia: *Lernen — Wikipedia, Die freie Enzyklopädie.* `http://de.wikipedia.org/w/index.php?title=Lernen\&oldid=74372925`. Version: 2010. – [Online; Stand 27. Juli 2010]

[52] Williams, Hywel ; Noble, Jason: Homeostatic Plasticity Improves Signal Propagation in Continous-Time Recurrent Neural Networks. In: *Biosystems* 87 (2006), S. 252 – 259

[53] Williams, Ronald J. ; Zipser, David: A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. In: *Neural Computation* 1 (1989), S. 270 – 280

Erklärung

Hiermit erkläre ich, die vorliegende Diplomarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet zu haben. Ich bin damit einverstanden, dass die vorliegende Arbeit in der Bibliothek des Instituts für Informatik der Humboldt-Universität zu Berlin öffentlich ausgelegt wird.

Berlin, den 20. September 2010

Matthias Kubisch