

Felix Schulte

Das Erlernen von Spielverhalten anhand des "Reinforcement Learning" bei Videospiele

Diplomarbeit

BEI GRIN MACHT SICH IHR WISSEN BEZAHLT



- Wir veröffentlichen Ihre Hausarbeit, Bachelor- und Masterarbeit
- Ihr eigenes eBook und Buch - weltweit in allen wichtigen Shops
- Verdienen Sie an jedem Verkauf

Jetzt bei www.GRIN.com hochladen
und kostenlos publizieren



Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de/> abrufbar.

Dieses Werk sowie alle darin enthaltenen einzelnen Beiträge und Abbildungen sind urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsschutz zugelassen ist, bedarf der vorherigen Zustimmung des Verlanges. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen, Auswertungen durch Datenbanken und für die Einspeicherung und Verarbeitung in elektronische Systeme. Alle Rechte, auch die des auszugsweisen Nachdrucks, der fotomechanischen Wiedergabe (einschließlich Mikrokopie) sowie der Auswertung durch Datenbanken oder ähnliche Einrichtungen, vorbehalten.

Impressum:

Copyright © 2007 GRIN Verlag
ISBN: 9783668626102

Dieses Buch bei GRIN:

<https://www.grin.com/document/388646>

Felix Schulte

Das Erlernen von Spielverhalten anhand des "Reinforcement Learning" bei Videospielen

GRIN - Your knowledge has value

Der GRIN Verlag publiziert seit 1998 wissenschaftliche Arbeiten von Studenten, Hochschullehrern und anderen Akademikern als eBook und gedrucktes Buch. Die Verlagswebsite www.grin.com ist die ideale Plattform zur Veröffentlichung von Hausarbeiten, Abschlussarbeiten, wissenschaftlichen Aufsätzen, Dissertationen und Fachbüchern.

Besuchen Sie uns im Internet:

<http://www.grin.com/>

<http://www.facebook.com/grincom>

http://www.twitter.com/grin_com

Danksagung

Mein besonderer Dank gilt Steffen Priesterjahn für die sehr gute Betreuung, die vielen (fachbezogenen) Gespräche und die zahlreichen Anregungen und Hilfestellungen. Es war mir eine große Freude, diese Arbeit über ein derart interessantes Thema und unter Verwendung des Klassikers *Quake* schreiben zu können.

Weiterhin danke ich meiner Frau Jasmin und meinem Sohn Max für die mentale und tatkräftige Unterstützung während des Schaffensprozesses.

Auch Bastian Schröder und Friedhelm Wegener von der Rechnerbetreuung haben mir große Dienste erwiesen. Ohne die zwei gestellten Rechner wäre das Experimentieren nicht unter diesen optimalen Bedingungen möglich gewesen.

Meinen Eltern gilt ebenfalls Dank für Hilfe, diese Arbeit sprachlich fehlerfrei zu halten.

Inhaltsverzeichnis

1	Einleitung	1
1.1	KI Forschung anhand von Videospiele	1
1.2	Thema der Arbeit	1
1.3	Warum Reinforcement Learning?	2
1.4	Ergebnisse der Arbeit	3
1.5	Aufbau der Arbeit	3
2	Verwandte Arbeiten	5
2.1	Evolutionäres Lernen im Spiel Quake3	5
2.2	Regelbasiertes System im Spiel Quake2	6
2.3	Navigational Behaviors im Spiel Half-Life	6
2.4	Support Vector Machines in Warcraft3	6
3	Reinforcement Learning	9
3.1	Einführung	9
3.2	Geschichte des Reinforcement Learnings	10
3.3	Grundlagen	10
3.3.1	Softwarekomponenten	10
3.3.2	Evaluative Feedback	16
3.3.3	Exploration vs. Exploitation	17
3.3.4	Nicht-stationäre Probleme	18
3.3.5	Periodische und kontinuierliche Aufgaben	18
3.3.6	Discounting	19
3.3.7	Markoveigenschaft	20
3.3.8	Backup-Diagramme	22
3.3.9	Bellmangleichung	22
3.4	Elementare Funktionen	26
3.4.1	Dynamic Programming	27
3.4.2	Monte-Carlo	31
3.4.3	Temporal Difference Learning	36

4	Entwicklung des Softwareagenten	47
4.1	Lernumgebung	47
4.2	Clientbot-API	49
4.3	Lernziele	51
4.4	Aufbau des Quakebots	51
4.5	Elemente des Reinforcement Learnings	54
	4.5.1 Q-learning	55
	4.5.2 Zustände und Aktionen	56
	4.5.3 Belohnungsfunktion	66
4.6	Statistik	67
4.7	Softwaredesign	68
	4.7.1 Entwurfsmuster	68
5	Experimente und Auswertungen	71
5.1	Einleitung	71
	5.1.1 Design of Experiments	72
	5.1.2 Vermeidung von Störfaktoren	73
5.2	Auswahl der Experimente	77
5.3	Durchführung der Experimente	79
5.4	Analyse der Experimente	79
	5.4.1 Abhängige Variable	79
	5.4.2 Darstellung der Ergebnisse	79
	5.4.3 Merkmale der Ergebnisse	81
	5.4.4 Experiment mit Standardeinstellung	82
	5.4.5 Variieren der Schrittgröße	82
	5.4.6 Variieren des Diskontierungsfaktors	85
	5.4.7 Variieren des Straffaktors	88
	5.4.8 Variieren des No-Hit Penaltys	90
5.5	Wissen des Agenten	92
	5.5.1 Der beste Zustand	92
	5.5.2 Die meist besuchten Zustände	93
	5.5.3 Zustandsübergänge	96
5.6	Co-Reinforcement Learning	97
6	Fazit	101
	Literatur	106

Abbildungsverzeichnis

3.3.1 Interaktion zwischen Agent und Umwelt (SB98, Seite 52)	11
3.3.2 Beispiel: Balancieren	20
3.3.3 Backup-Diagramm: Zustands-Wertefunktion	23
3.3.4 Backup-Diagramm: Aktions-Wertefunktion	25
3.3.5 Backup-Diagramm: optimale Wertefunktion	26
3.4.1 Policy Iteration	31
3.4.2 Backup-Diagramm: Monte-Carlo	34
3.4.3 Generalized Policy Iteration: MC	35
3.4.4 Backup-Diagramm: Temporal Difference	38
3.4.5 Beispiel: Zeitschätzung MC	40
3.4.6 Beispiel: Zeitschätzung TD	40
3.4.7 Beispiel: Markov-Prozessmodell	42
3.4.8 Backup-Diagramm: Q-learning	45
3.4.9 Gridworld mit Klippe (SB98, Seite 150)	46
4.1.1 Quake3 Spielszene	48
4.1.2 Quake, Clientbot, Agent	49
4.1.3 Lernumgebung	50
4.4.1 Klassendiagramm	52
4.4.2 Interaktion: Agent – Spielumgebung	53
4.4.3 IOController	55
4.5.1 Erstellen der Zustandsmatrix	58
4.5.2 Matrix mit Gaußfilter überarbeitet	59
4.5.3 Drei Zustände	60
4.5.4 Matrizen mit Gaußfilter überarbeitet	60
4.5.5 Basisaktionen	65
4.5.6 Überarbeitete Blickrichtungsänderungen	65
5.1.1 Versuchsaufbau	72
5.4.1 Beispiel: Versuchsergebnisse	80
5.4.2 Variation von α	83

5.4.3 Variation von α (mit Standardabweichung)	84
5.4.4 Aktionswertänderung	85
5.4.5 Variation von γ	86
5.4.6 Variation von γ (mit Standardabweichung)	87
5.4.7 Variation des Penalty Factors	88
5.4.8 Variation des Penalty Factors (mit Standardabweichung) . . .	89
5.4.9 Variation des No-Hit Penaltys	91
5.5.1 Bester Zustand	92
5.5.2 Meist besuchter Zustand	93
5.5.3 Zweitmeist besuchter Zustand	94
5.5.4 Drittmeist besuchter Zustand	95
5.5.5 Zustandsübergangsfunktion	96
5.6.1 Gegeneinander spielende Bots	98
5.6.2 Gegeneinander spielende Bots	99

Kapitel 1

Einleitung

1.1 KI Forschung anhand von Videospielen

Videospiele fungieren als beliebte Plattform für die Entwicklung und Erforschung künstlicher Intelligenz (KI). Sie stellen ein geeignetes Thema für akademische Forschung dar (Lai01). Videospiele sind anspruchsvoll und gleichzeitig recht einfach zu formalisieren (MBC⁺06, Seite 151). Sie bieten komplexe und realitätsnahe Umgebungen (Lai01). Spielzustände und Verhaltensweisen der nicht-deterministischen Umwelt sind nicht vorhersagbar – optimale Verhaltensweisen sind nicht bekannt. Der Entwickler kann deshalb dem Softwareagenten nur in begrenztem Maße Wissen zur Verfügung stellen. Der Agent muss Verhaltensweisen selbstständig erlernen. Videospiele machen es möglich, auf einfache Weise Methoden der künstlichen Intelligenz zu entwickeln und ihre Verhaltensweisen zu analysieren (MBC⁺06, Seite 151).

Intelligente Agenten können das Verhalten ihrer Umwelt erlernen und so auf verschiedene Situationen angemessen reagieren. Sie sind weiterhin in der Lage, sich auf neue Situationen einzustellen. Diese Eigenschaft ist von großer Bedeutung in der Entwicklung von Robotern in menschlicher Umgebung (MBC⁺06).

Videospiele bieten also nicht nur eine leicht formalisierbare und messbare Umgebung für eine künstliche Intelligenz – sie gewährleisten einen Entwicklungsprozess ohne Gefährdung von Lebewesen oder Eigentum.

1.2 Thema der Arbeit

Laut Aufgabenstellung soll ein Softwareagent entwickelt werden, welcher sich Spielverhalten unter Verwendung von Reinforcement Learning aneignet. Die Aufgabenstellung sieht weiterhin die Verwendung bestimmter Techniken vor.

Die Wahrnehmung der Umwelt erfolgt mithilfe eines sogenannten *Grids*. Ein Grid ist eine Matrix, in welcher die Beschaffenheit der Umgebung kodiert ist. Diese Technik wurde von Steffen Priesterjahn et al. für eine dem Thema dieser Diplomarbeit ähnlichen Arbeit entwickelt (siehe Abschnitt 2.1 und (PKWB06)). Ich stelle diese Technik in Abschnitt 4.5.2 ausführlich vor.

Desweiteren sollte der k -Means-Algorithmus zum Einsatz kommen, um die Zustandsmenge auf ein geeignetes Maß zu reduzieren. Genaue Erläuterungen zu k -Means und Clusterverfahren im Allgemeinen finden sich ebenfalls in Abschnitt 4.5.2.

1.3 Warum Reinforcement Learning?

Zum Zeitpunkt der Entwicklung des Agenten ist nicht bekannt, welches Spielverhalten des Agenten optimal ist. Das vom Agenten verwendete Lernverfahren muss also zu den unüberwachten Verfahren gehören.

In Abschnitt 2.1 stelle ich das Konzept eines Quakebots vor, in welchem ein unüberwachtes Lernverfahren – ein evolutionärer Algorithmus – verwendet wird. In Anlehnung an diese Arbeit entstand die Idee, einen Agenten auf der Basis des *Reinforcement Learning* (deutsch sinngemäß *verstärkendes Lernen*) zu entwickeln.

Reinforcement Learning bietet einige Vorteile gegenüber evolutionären Lernverfahren. Reinforcement Learning beschreibt Lernen zum Zeitpunkt der Interaktion mit der Umgebung, während evolutionäre Verfahren zeitlich versetzt zur Interaktion lernen. R. Sutton und A. Barto sind davon überzeugt, dass Methoden, welche die Vorteile des Verhaltens einzelner Individuen nutzen können, in vielen Fällen effizienter lernen können als evolutionäre Methoden (SB98, Seite 9). Evolutionäre Lernverfahren ignorieren Informationen über die Struktur des Lernproblems. Sie erkennen nicht, dass die gesuchten Strategien Funktionen von Zuständen zu Aktionen sind. Sie betrachten weder die Zustände noch die Aktionen, welche von den Individuen während ihrer Interaktion mit der Umgebung durchlaufen werden. Durch das Verwenden von zusätzlichem Wissen des Individuums sollte in den meisten Fällen eine effizientere Suche nach guten Strategien möglich sein (SB98, Seite 9).

Es gibt auch gegensätzliche Meinungen zum Einsatz von Reinforcement Learning in Videospielen. Um Adaption in Videospielen realisierbar zu machen, benötigt ein Lernverfahren Eigenschaften wie Leistungsfähigkeit, Flexibilität und Zuverlässigkeit. Reinforcement Learning weist diese Attribute zwar in wenig komplexen Bereichen auf, sei aber für Aufgaben mit der Komplexität von Videospielen ungeeignet (MBC⁺06, Seite 157).