hochschule mannheim

Konstruktion eines Linienfolgers mittels Q-Learning

Minh Le, Nicolas Pilch, Stefan Witsuba

Ausarbeitung

für die Mündliche Prüfung in KIS

Studiengang Informatik

Fakultät für Informatik

Hochschule Mannheim

17.07.18

Professor

Prof. Jörn Fischer, Hochschule Mannheim

Abstract

Konstruktion eines Linienfolgers mittels Q-Learning

Jemand musste Josef K. verleumdet haben, denn ohne dass er etwas Böses getan hätte, wurde er eines Morgens verhaftet. Wie ein Hund! sagte er, es war, als sollte die Scham ihn überleben. Als Gregor Samsa eines Morgens aus unruhigen Träumen erwachte, fand er sich in seinem Bett zu einem ungeheueren Ungeziefer verwandelt. Und es war ihnen wie eine Bestätigung ihrer neuen Träume und guten Absichten, als am Ziele ihrer Fahrt die Tochter als erste sich erhob und ihren jungen Körper dehnte. Es ist ein eigentümlicher Apparat, sagte der Offizier zu dem Forschungsreisenden und überblickte mit einem gewissermaßen bewundernden Blick den ihm doch wohl bekannten Apparat. Sie hätten noch ins Boot springen können, aber der Reisende hob ein schweres, geknotetes Tau vom Boden, drohte ihnen damit und hielt sie dadurch von dem Sprunge ab. In den letzten Jahrzehnten ist das Interesse an Künstlern sehr zurückgegangen. Aber sie überwanden sich, umdrängten den Käfig und wollten sich gar nicht fortrühren.

Inhaltsverzeichnis

1 Einleitung **1**

2 Robotik **2**
2.1 Linienfolger 2
2.2 Lego Mindstorm EV3 2

3 Machinelles Lernen in der Robotik **3**
3.1 Reinforcement Learning 3
3.2 Q-Learning 4

4 Implementierung **6**

5 Evaluation **7**

6 Fazit **11**

Abkürzungsverzeichnis **ii**

Tabellenverzeichnis **ii**

Abbildungsverzeichnis **iii**

Quellcodeverzeichnis **iv**

Literatur **vi**

Kapitel 1

Einleitung

Einleitung

Kapitel 2

Robotik

Lego Roboter

2.1 Linienfolger

Die Idee des Linienfolgers stammt von der Rescue-League des Robocups, einem Wettbewerb in dem es ursprünglich um einen Roboterfußball-Wettkampf ging. Bei der Rescue League müssen die Roboter einer Linie folgen und am Ende ein „Opfer“ erkennen und bergen. Erschwert wird dies durch Rampen, Lücken und Hindernisse auf und neben der Linie.

In vereinfachter Form geht es zunächst nur um die Linie und so muss der Robotor einfach einer schwarzen Linie folgen.

2.2 Lego Mindstorm EV3

leJOS ist die nunmehr dritte Generation des Java-Betriebssystems für den programmierbaren Lego-Mindstorms Stein EV3. Diese Software erlaubt die Steuerung von diversen Lego-Konstruktionen in Java. Unter anderem können so verschiedene Sensoren ausgelesen und Motoren angesteuert werden.

50: 00: G01: L10: G11: L 100:00: R01: L10: R11: L 150:00: L01: L10: G11: L 200:00: G01: L10: R11: L 250:00: G01: L10: R11: R 300:00: L01: L10: G11: R 350:00: G01: L10: L11: R 400:00: G01: L10: L11: R 450:00: G01: L10: L11: R 500:00: G01: G10: R11: R 550:00: G01: L10: R11: R

Kapitel 3

Machinelles Lernen in der Robotik

Die Robotik ist ein klassisches Aufgabenfeld des Machinellen Lernen. Da die durchzuführenden Aufgaben meist zu komplex sind und für diese daher keine Trainingsdaten zur Verfügung stehen, muss der Roboter durch Ausprobieren und dem daraus resultierenden Ergebnis die optimale Strategie zur behebung des Problems finden. Durch diese Problemstellung eignet sich die Robotik insbesonders für den Einsatz von Lernen durch Verstärkung. Ertel 2013

3.1 Reinforcement Learning

Lernen durch Verstärkung, bzw. Reinforcement Learning, beschreibt ein Lernverhalten für das keine Trainingsdaten oder ähnliches vorliegen. Der Lernende bzw. Agent kann sein Verhalten nur durch Untersuchung des von der Umgebung auf eine seiner Handlungen zurückgegebenen Feedbacks optimieren. Durch das Ausführen einer Aktion wechselt, oder bleibt, der Agent in einem Zustand. Durch den nun angenommenen Zustand wird eine Belohnung definiert, die die Güte der soeben ausgeführten Aktion unter Einbeziehung des Ausgangszustandes wiederspiegelt. Ertel 2013

Der Zustand $s_t \in S$ beschreibt die Umgebung des Roboters und ihn selbst zu einem bestimmten Zeitpunkt. Dies geschieht mit einer gewissen Abstraktion, da die Realität meist zu komplex ist um diese genau abzubilden und dem Agenten durch Messungenauigkeiten oder ähnliches die Informationen fehlen. Aus dem vorliegenden Zustand wird die passende Aktion $a_t \in A$ durch eine Strategie ausgewählt. Diese

wird anschließend ausgeführt und die von der Umgebung zurückgelieferte Reaktion bzw. reward untersucht. Durch diesen wird die Strategie optimiert. Ertel 2013

Die Strategie $\pi : S \rightarrow A$ ist optimal, wenn langfristig ein maximaler Ertrag erreicht wird. Der Wert bzw. die Belohnung der optimale Strategie wird durch die Bewertungsfunktion

$$V^{\pi}(s_t) = \sum_{i=0}^{\infty} y^i r_{t+1} \tag{3.1}$$

beschrieben, in der direkte Belohnungen durch den Faktor $0 \leq y < 1$ stärker einfließen sollen als weiter in der Zukunft liegende. Durch dieses kann der Agent in jedem Zustand die optimale Aktion auswählen. Ertel 2013

Dies führt in der Robotik zu Problemen, da meist nicht klar ist, was für ein Zustand nach der Ausführung einer Aktion angenommen wird. Da der Wert der Strategie auf die Bewertung der Nachfolgezustände basiert kann dies nicht angewendet werden. Als Lösung dieses Problems nutzt man Q-Learning.

3.2 Q-Learning

Beim Q-Learning wird eine Bewertungsfunktion $Q(s, a)$ eingeführt. Die optimale Aktion zu einem gegebenen Zustand wird nun durch die maximale Bewertungsfunktion für den Zustand definiert. Durch Umformung der Gleichung 3.1 wird der Wert der optimalen Bewertungsfunktion mit

$$Q_t^* = r_t + y_* Q_{t+1}^* \tag{3.2}$$

definiert. Der Wert der aktuellen Bewertungsfunktion ist sommit die Belohnung des aktuellen Zustandes und der Wert der abgeschwächten Nachfolgezustandsbewertungsfunktionen. Des Weiteren benutzt man meist noch eine Lernrate α:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_t + y\ max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{3.3}$$

Ertel 2013

Zu Anfang werden die Werte der Bewertungsfunktionen zufällig initialisiert. Für den aktuellen Zustand wird die Aktion mit der größten Bewertungsfunktion ausgeführt und der neue Zustand und die daraus resultierende Belohnung analysiert.

Der Wert der Bewertungsfunktion $Q(a, b)$ wird aktualisiert. Anschließend wird mit dem neuen Zustand wie mit dem vorherigen Verfahren. Dies wiederholt sich bis ein Endzustand erreicht worden ist. Ertel 2013

Kapitel 4

Implementierung

Kapitel 5

Evaluation

Für die Erprobung des Q-Learning-Algorithmuses wurden zwei Linienfolger und eine Teststrecke konzipiert und zusammengebaut. Der Kurs (siehe Abbildung 5.1) besteht aus verschiedenen Geraden- und Kurvenstücken von variablem Schwierigkeitsgrad.

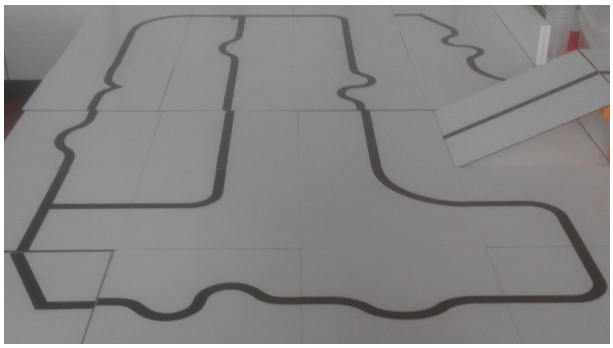

Abbildung 5.1: Kurs

Die beiden Linienfolger unterscheiden sich in der Anzahl und der Anordnung der Farbsensoren. Der erste Linienfolger L_0 verfügt über drei Farbsensoren, die nebeneinander angeordnet sind. Die schwarze Linie besitzt dieselbe Breite wie der Lichtkegel eines Farbsensors. Sommit ist der Linienfolger optimal Positioniert falls nur der mittlere Sensor Schwarz und die beiden außenliegenden Sensoren Weiß erkennen. Die Tabelle 5.1 zeigt alle eintretbaren Zustände.

Im Gegensatz zum ersten Linienfolger besitzt der zweite Linienfolger L_1 zwei Farbsensoren, die angewinkelt angeordnet sind. Sommit besitzt er andere Zustände als L_0, diese sind in der Tabelle 5.2 beschrieben.

Tabelle 5.1: Zustände bei drei Sensoren

Zustand	Beschreibung	Belohnung
s_0	Alle Sensoren erkennen Weiß.	0
s_1	Der rechte Sensor erkennt Schwarz.	0
s_2	Der mittlere Sensor erkennt Schwarz.	0
s_3	Der mittlere und rechte Sensor erkennt Schwarz.	0
s_4	Der linke Sensor erkennt Schwarz.	0
s_5	Der linke und rechte Sensor erkennt Schwarz.	0
s_6	Der linke und mittlere Sensor erkennt Schwarz.	0
s_7	Alle Sensoren erkennen Schwarz.	0

Tabelle 5.2: Zustände bei zwei Sensoren

Zustand	Beschreibung	Belohnung
s_0	Beide Sensoren erkennen Weiß.	0
s_1	Der rechte Sensor erkennt Schwarz.	0
s_2	Der linke Sensor erkennt Schwarz.	0
s_3	Beide Sensoren erkennen Schwarz.	1

Zur Erreichung eines anderen Zustandes können beide Linienfolger folgende Aktionen (siehe Tabelle 5.3) ausführen. Die Art der Ausführung unterscheidet sich nicht bei L_0 oder L_1.

Tabelle 5.3: Aktionen

Aktion	Beschreibung
a_0	Linkskurve
a_1	Geradeausfahrt
a_2	Rechtskurve

Beide Linienfolger nutzen für das Q-Learning die gleichen Parameter, um die Vergleichbarkeit der Ergebnisse zu gewährleisten.

Die optimale Strategie für L_0 ist im Graph 5.2 angegeben, die für L_1 in Graph 5.3.

Bei beiden Linienfolgern wurden die optimalen Aktionen für alle Zustände alle 50 Lernschritte protokoliert.

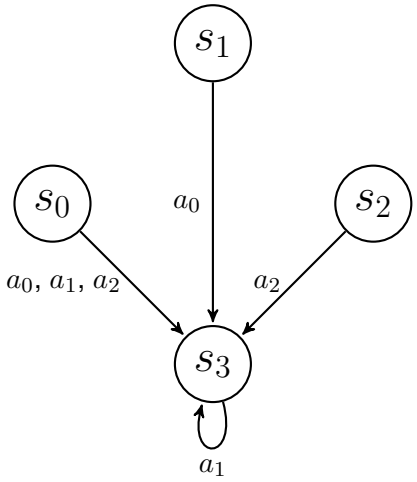

Abbildung 5.2: Optimale Strategie bei zwei Sensoren

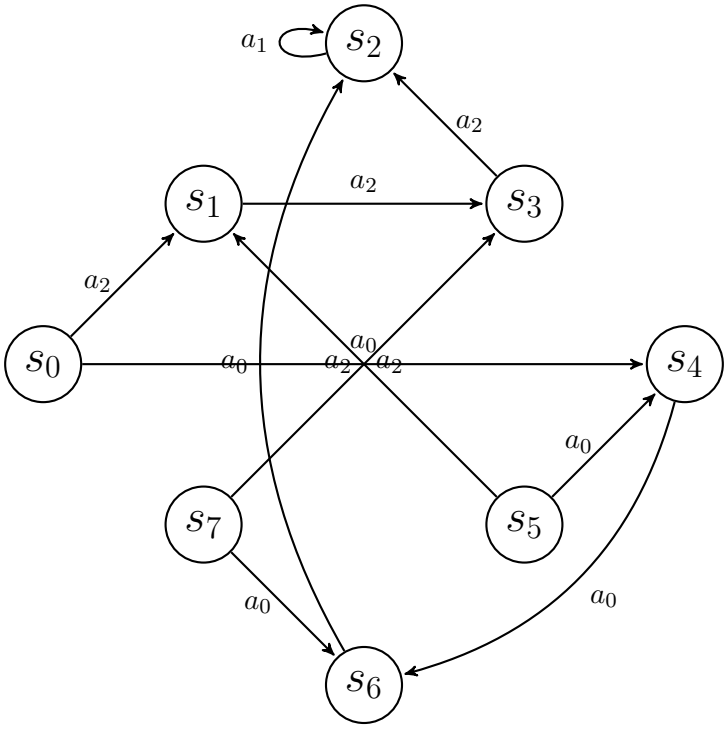

Abbildung 5.3: Optimale Strategie bei drei Sensoren

Tabelle 5.4: Q-Learning-Parameter

Parameter	Wert
α	0.9
γ	0.2
ϵ	0.01

Kapitel 6

Fazit

Fazit

Abkürzungsverzeichnis

Tabellenverzeichnis

5.1 Zustände bei drei Sensoren 8
5.2 Zustände bei zwei Sensoren 8
5.3 Aktionen 8
5.4 Q-Learning-Parameter 10

Abbildungsverzeichnis

5.1 Kurs 7
5.2 Optimale Strategie bei zwei Sensoren 9
5.3 Optimale Strategie bei drei Sensoren 9

Listings

Literatur

Ertel, Wolfgang (2013). *Grundkurs Künstliche Intelligenz*. Springer Fachmedien Wiesbaden. DOI: 10.1007/978-3-8348-2157-7.