Lego Mindstorms Vier-Gewinnt-Roboter

Robert Göpfert, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—In diesem Paper wird die Entwicklung eines Roboters mit LEGO Mindstorms beschrieben, welcher autonom Vier Gewinnt gegen einen menschlichen Gegner spielen kann. Dafür wurde aufbauend auf der mathematischen Spieltheorie ein einfacher Spielalgorithmus implementiert. Die Spielfelderkennung wurde weiterhin durch die Verwendung einer Webcam automatisiert.**

***Schlagwörter*—Künstliche Intelligenz, Minimax-Algorithmus, Projektseminar, Robotik, Spieltheorie, Vier Gewinnt**

I. Einleitung

EIN stark wachsendes Teilgebiet der Informatik ist die Entwicklung künstlicher Intelligenz, um Mustererkennung und -vorhersage und logische Schlüsse basierend auf einer Wissensdatenbank zu automatisieren. Anwendungsgebiete sind die Erkennung von Krankheitsbildern in der Medizin, Bild- und Gesichtserkennung in der Kriminalistik und die Nachbildung menschlichen Verhaltens von Computergegnern in Videospielen. Zu Computerspielen zählen auch Brettspiele, wie Solitär, Schach und das in diesem Paper behandelte Vier Gewinnt. Bezüglich des Projektseminars wurde dafür mit LEGO Mindstorms ein Roboter entwickelt, der ein physisches Spielfeld mit einer Webcam erkennt und basierend auf diesen Informationen den nächsten Zug berechnet und an entsprechender Stelle seinen Spielstein einwirft.

II. Vorbetrachtungen

Um eine künstliche Intelligenz für ein Spiel zu entwickeln, werden Ansätze der Spieltheorie benutzt. Dazu wird das Spiel zunächst näher beschrieben.

A. Vier Gewinnt

Vier Gewinnt wird zu zweit auf einem senkrecht stehenden Spielbrett gespielt. Ziel ist es, als erster 4 eigene Spielsteine in eine Reihe horizontal, vertikal oder diagonal in eine Reihe zu bringen. Dabei handelt es sich bei Vier Gewinnt um ein endliches Nullsummenspiel mit perfekter Information. Endlich heißt, dass nach einer endlichen Anzahl an Zügen das Spiel endet. Ein Nullsummenspiel ist ein Spiel, bei dem die Summe der Verluste und Gewinner beider Spieler zusammengenommen gleich null ergeben. [1] Bei einem Spiel mit perfekter Information lässt sich der Spielstand jederzeit von allen Spielteilnehmern einsehen. In Abbildung 1 ist eine Spielsituation dargestellt, in der der Spieler mit den gelben Steinen gewinnt.

DOI: 10.24352/UB.OVGU-2019-040

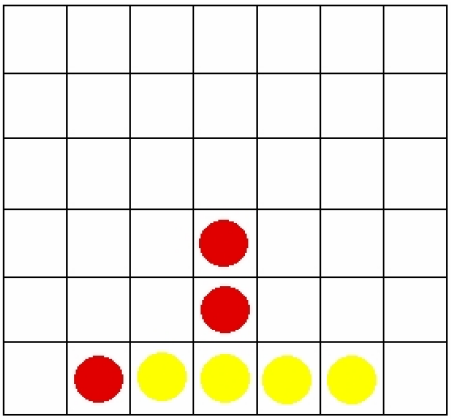

Abbildung 1. Beispielhafte Spielsituation mit Gewinn für Gelb, Quelle: urlhttp://www.mathematik.uni-muenchen.de/~spielth/artikel/VierGewinnt.pdf

B. Bewertungsfunktion

Um einen Spielalgorithmus für Vier Gewinnt zu implementieren, ist eine Bewertungsfunktion notwendig, mit der es möglich ist, eine Spielsituation nach den Gewinnchancen für die jeweiligen Spieler zu evaluieren. Eine ideale Bewertungsfunktion stellt die Funktion 1 dar.

$$f(M) = \begin{cases} 1 & \text{Spieler gewinnt} \\ 0 & \text{Unentschieden} \\ -1 & \text{Gegenspieler gewinnt} \end{cases} \quad (1)$$

Die Verwendung einer idealen Bewertungsfunktion ist allerdings nur bei einfachen Spielen wie Tic-Tac-Toe angebracht. Für Spiele wie Schach oder Vier Gewinnt existieren zu viele verschiedene Kombinationen, um all diese in einer annehmbaren Zeit zu bearbeiten. Dafür wird die Bewertungsfunktion wie in Funktion 2 angepasst.

$$f(M) = \begin{cases} \infty & \text{Spieler gewinnt} \\ n \in \mathbb{N} > 0 & \text{Vorteil für Spieler} \\ 0 & \text{kein Vorteil für beide Spieler} \\ n \in \mathbb{N} < 0 & \text{Vorteil für Gegenspieler} \\ -\infty & \text{Gegenspieler gewinnt} \end{cases} \quad (2)$$

Dabei stehen negative Werte für einen Vorteil für den Gegenspieler und positive für den Spieler. Je größer ihr Betrag ist, desto größer ist der Vorteil für den jeweiligen Spieler. ∞ steht für den Gewinn des jeweiligen Spielers.

Eine Bewertungsfunktion speziell für Vier Gewinnt könnte so gestaltet werden, dass höhere Werte zurückgegeben werden, je vollständiger eine Reihe ist. Außerdem könnten Spielsituationen

mit Zwickmühlen bevorzugt werden, indem sie höher als andere Vervollständigungen bewertet werden. Zwickmühlen sind bei Vier Gewinnt Situationen, die zwei unvollständige sich überkreuzende Reihen enthalten, die nicht gleichzeitig blockiert werden können.

C. Minimax-Algorithmus

Der Minimax-Algorithmus wird eingesetzt, um die optimale Strategie für ein endliches Nullsummenspiel mit perfekter Information, wie Vier Gewinnt, zu ermitteln. Allgemein gesagt werden ausgehend von der aktuellen Spielsituation alle möglichen zukünftigen Szenarien bestimmt und nach der Gewinnchance für den Spieler bewertet. Danach wird eine Strategie ausgearbeitet, um die Spielsituation mit der höchsten Gewinnchance zu sichern.

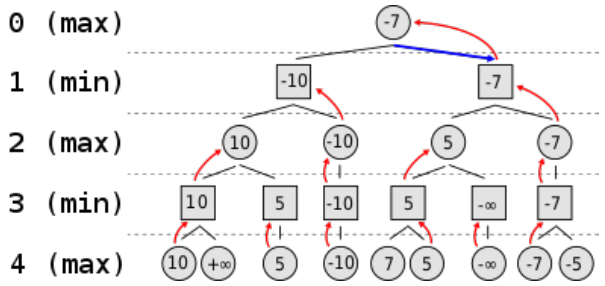

Abbildung 2. Suchbaum eines Minimax-Algorithmus, Quelle: https://upload.wikimedia.org/wikipedia/commons/6/6f/Minimax.svg

Um alle künftigen Spielsituationen abzudecken, wird rekursiv ein Suchbaum aufgebaut. In Abbildung 2 ist ein Suchbaum mit der Tiefe 4 dargestellt. Die aktuelle Spielsituation bildet den Wurzelknoten des Baumes. Alle davon abgehenden Kindknoten enthalten unterschiedliche Spielsituation nach einem Zug des Gegners. Ist die maximale vorgegebene Tiefe erreicht, wird auf die Spielsituationen der untersten Blätter eine Bewertungsfunktion wie in Abschnitt II-B beschrieben angewendet.

Anschließend wird in den Ebenen, in denen der Gegenspieler am Zug ist, die kleinsten Werte aus den Kindknoten ausgewählt und den Elternknoten zugewiesen (es wird minimiert). In den Ebenen, in denen der Spieler am Zug ist, werden die größten Werte aus den Kindknoten ausgewählt und den Elternknoten zugewiesen (es wird maximiert). Ist der Wurzelknoten erreicht, wird aus dessen Kindknoten der größte Werte ausgewählt. Dieser Knoten ist dann auch der Zug der gespielt wird. [2], [4]

III. Hauptteil

A. Erkennung des Spielfeldes

Der erste Ansatz zur Spielfelderkennung bestand darin, mit einem Farbsensor die einzelnen Zellen zu scannen und anhand der Farbwerte und der gescannten Position eine Matrix aufzubauen. Um Zeit einzusparen, sollten dabei die Zellen, die bereits gescannt wurden, übersprungen werden. Diese Variante der Spielfelderkennung war jedoch trotz der Optimierung zu zeitintensiv, da im ungünstigsten Fall sieben Zellen pro Scan untersucht werden müssen.

Als effizientere Variante stellte sich die Verwendung einer Webcam heraus, da mit dieser das gesamte Spielfeld auf einmal eingescannt werden konnte. Probleme bereitete allerdings die Erkennung der Position und Farbe der einzelnen Zellen. Anfangs wurde die in Matlab integrierte Funktion „imfindcircles“ verwendet, welche kreisförmige Objekte in einem Bild erkennen kann.

Da diese jedoch unzuverlässig arbeitete, wurden die Positionen der Zellen vorher festgelegt, sodass nur noch die Ausrichtung der Kamera eine Fehlerquelle darstellt. Anschließend wurden die Farbwerte an den entsprechenden Positionen ausgelesen, einer Zahl zugeordnet und in die Matrix eingetragen.

B. Ermittlung des besten Zuges

Nachdem das Spielfeld in eine Matrix übertragen wurde, bewertete eine Prüfroutine die aktuelle Spielsituation. Eine solche ist mit Funktion 3 dargestellt. Sie diente dazu, das Spiel zu beenden, wenn ein Gewinner feststand, es unentschieden stand oder geschummelt wurde. Der Algorithmus erkennt das mehrfache Einwerfen während eines Zuges und das Entfernen von Spielsteinen sowie das Auslassen eines Einwurfes als Schummeln und beendet bei entsprechenden Aktionen das Spiel.

$$f(M) = \begin{cases} 0 & \text{, noch kein Ergebnis} \\ 1 & \text{, rot hat gewonnen} \\ 2 & \text{, gelb hat gewonnen} \\ 3 & \text{, unentschieden} \\ 4 & \text{, es wurde geschummelt} \end{cases} \tag{3}$$

Da sich die Arbeit am Bau des Roboters nur auf zwei Wochen beschränkte, wurde ein sehr einfacher Algorithmus geschrieben. Dieser priorisiert die Vervollständigung einer eigenen Reihe vor der Blockierung einer gegnerischen Reihe. Kann weder vervollständigt noch blockiert werden, wird an zufälliger Stelle ein Stein eingeworfen.

C. Anfahren der Position und Einwurf des Steins

Die Grundidee des Aufbaus bestand darin, die Spielsteine aus einer erhöhten Position in das Spielbrett einzuwerfen. Daher wurde ein Turm konstruiert, der sich auf vier Rädern parallel zum Spielfeld bewegen kann, wie in Abbildung 3 zu sehen ist. An der Turmspitze schob ein mechanischer Arm die Steine aus einem Vertikalmagazin auf eine geneigte Ebene, auf der sie dann in die Öffnungen des Spielbrettes rutschten. Um einen konstanten Abstand zum Brett sicherzustellen, wurden an das Ende der Rutsche Leitschienen angebracht. Seitlich am Turm befand sich weiterhin ein Taster, mit dem der menschliche Gegenspieler signalisieren konnte, dass er seinen Zug beendet hat.

Der Roboter wurde anfangs so ausgerichtet, dass das Ende der Rutsche über dem Brett positioniert war und ein aus dem Magazin geschobener Stein in die erste Spalte fallen würde. Von dort aus wurden die restlichen Spalten angefahren.

Nachdem der Roboter die gewünschte Position angefahren hatte, erfolgte der Einwurf eines Steines und der Roboter fuhr zweimal eine kleine Strecke nach links und rechts, um eine mögliche Verkantung des Steines mit dem Spielbrett zu lösen.

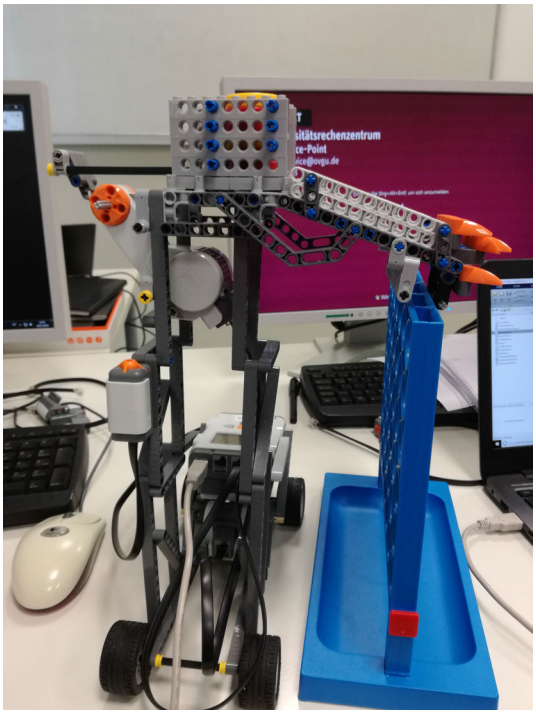

Abbildung 3. Am Spielbrett ausgerichteter und einsatzbereiter Vier-Gewinnt-Roboter

D. Gesamtes Programm

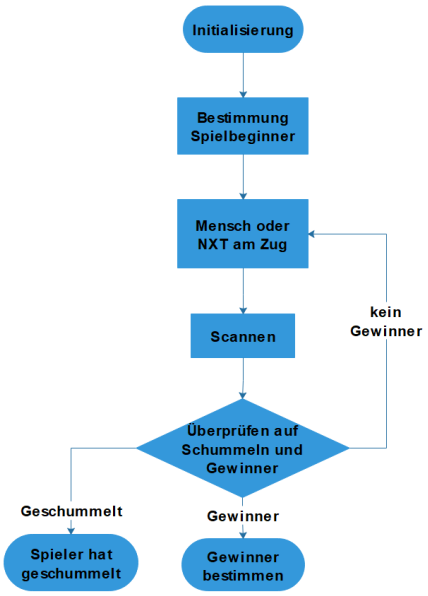

Abbildung 4. Gesamter Programmablaufplan des Vier-Gewinnt-Roboters

In Abbildung 4 ist der Programmablaufplan des gesamten Programms abgebildet. Nachdem die Kamera ausgerichtet wurde, konnte das Spiel über die graphische Benutzeroberfläche gestartet werden. Über diese wurde ausgegeben, welcher Spieler beginnt. Begann der menschliche Spieler, führte dieser seinen Zug durch und betätigte anschließend den Taster am Roboter. Daraufhin wurde das Spielfeld gescannt und überprüft, ob geschummelt wurde und ob ein Spieler das Spiel gewonnen hat oder es unentschieden ist. Traf eines der Kriterien zu, wurde das Spiel beendet und eine entsprechende Meldung über die Benutzeroberfläche ausgegeben. Wenn nicht, wurde das Spiel fortgesetzt, indem der Roboter seinen Zug ausführte, das Spielfeld scannt und auf die Betätigung des Tasters wartet.

IV. Ergebnisdiskussion

Während des Projektseminars wurde ein Roboter entwickelt, welcher autonom gegen einen menschlichen Spieler Vier Gewinnt spielen konnte. Aufgrund der zeitlichen Begrenzung wurde nur ein einfacher Spielalgorithmus verwendet, gegen den der menschliche Spieler relativ einfach gewinnen konnte. Dies bestätigten auch mehrere Testspiele, in denen der Roboter nur gewann, wenn die Testperson ihm die Gewinnchancen nicht verbaute. Weiterhin beendet der Algorithmus das Spiel automatisch, wenn ein Gewinner feststeht, es unentschieden steht oder geschummelt wurde.

Ein Problem stellte anfangs der Einwurf der Spielsteine dar. Trotz der relativ präzisen Rutschenkonstruktion kam es oft vor, dass sich die Spielsteine mit dem Rand des Spielbrettes verkanteten. Dies konnte gelöst werden, indem der innere Rand der Öffnungen trichterartig angeschliffen wurde, sodass die Steine direkt in die Öffnung hineinfielen. Weiterhin kam es häufig zum gleichzeitigen Einwurf mehrerer Spielsteine, wenn sich nur noch zwei oder drei Steine im Magazin befanden. Dieses Problem konnte nicht gelöst werden, man konnte nur dafür sorgen, dass immer genug Steine nachgelegt wurden. Auch die Erkennung der Spielsituation mittels Webcam verlief nicht ohne Probleme. Da anfangs die relativ ungenaue Matlab-Funktion „imfindcircles“ verwendet wurde, musste dabei auf eine optimale Belichtung des Spielfeldes geachtet werden. Auch mit der finalen Lösung durch die Festlegung der Zellenpositionen musste auf die exakte Ausrichtung der Webcam und halbwegs gute Lichtverhältnisse geachtet werden. Dies sorgte zum einen dafür, dass der anfängliche Kalibrierungsvorgang durch mehrere Versuche länger dauerte, zum anderen durften das Spielfeld und die Webcam während des Spielvorgangs nicht gegeneinander verschoben werden.

V. Zusammenfassung und Fazit

Während des Projektseminars gelang es, einen Roboter zu bauen, der in der Lage war, Vier Gewinnt gegen einen menschlichen Gegner zu spielen. Zukünftige Verbesserungen finden sich vor allem in der Programmierung. Zum einen kann die Spielfelderkennung mittels Webcam verbessert werden, indem farbige Markierungen an die Ecken des Spielfeldes angebracht werden. Aus den Positionen dieser Marker können dann die Positionen der Zellenmittelpunkte bestimmt werden. Somit wäre die Spielfelderkennung unabhängig von der Positionierung und vom Ausrichtungswinkel der Webcam und die zeitintensive Kalibrierung zu Beginn eines Spiels würde entfallen. Zum anderen kann der Spielalgorithmus zum Minimax-Algorithmus abgeändert werden, sodass der Roboter intelligentere Züge spielen kann. Weiterhin kann für die Farbbestimmung einer Zelle ein Mittelwert aus den Farbwerten mehrerer Pixel gebildet werden. Somit würde man aussagekräftigere Werte

erhalten und die Spielfelderkennung etwas unabhängiger von der Umgebungshelligkeit machen.

Außerdem kann das Magazin und der Einwurfmechanismus verbessert werden, sodass es bei wenigen Spielsteinen nicht mehr zu Verkantungen und mehrfachem Einwurf pro Zug kommen kann.

LITERATUR

[1] Wikipedia, Nullsummenspiel, https://de.wikipedia.org/wiki/Nullsummenspiel (Abruf: 19.03.2019)
[2] Wikipedia, Minimax-Algorithmus, https://de.wikipedia.org/wiki/Minimax-Algorithmus (Abruf: 19.03.2019)
[3] Christian Kanzow und Alexandra Schwartz *Spieltheorie: Theorie und Verfahren zur Lösung von Nash- und verallgemeinerten Nash-Gleichgewichtsproblemen, S.5-7*. 02.Oktober 2018.
[4] Hrsg. v. Görz, Günther / Schneeberger, Josef / Schmid, Ute *Handbuch der Künstlichen Intelligenz, S.624-627*. Oktober 2013.