’Emi.mem’ – MemoryBot

Tim Jan Müller, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—In diesem Paper soll es spezifisch um die Schwierigkeiten und Herausforderungen gehen, die es bei der Umsetzung der Idee gab, einen Memory spielenden Roboter zu entwerfen, der sowohl allein, als auch als Spielpartner fungieren soll. Diesbezüglich werden im folgenden sowohl verworfene Ideen und Konzepte, als auch das abgeschlossene Modell vorgestellt und detailliert erläutert.**

***Schlagwörter*—FEIT, LEGO-Mindstorm, Memory, Roboter**

I. Einleitung

In vielen Bereichen des alltäglichen Lebens werden Prozesse durch autonom fungierende Robotoren erleichtert. Ob zu Hause, wie Staubsaug- und Kochroboter (z.B. ThermoMix), oder in der Industrie, als Hilfe bei der Fließbandarbeit, maschinelle Hilfsmittel sind nicht mehr wegzudenken. Jedoch steckt hinter diesen meist kompakten Anlagen eine Menge Programmier- und Konstruktionsarbeit.
Um den Studenten der FEIT dieses Thema näher zu bringen, findet in unserer Universität seit 8 Jahren das LegoMindstorms-Projekt statt. Die Studenten können sich hier am Entwurf, der Konstruktion und der Programmierung eines eigenen Roboters testen. Die einzige Einschränkung war dabei, dass dieser aus dem vorhandenen Material des Lego-Mindstorm-Set bestehen soll. Der Zeitraum, in dem das Projekt stattfand, lag bei 2 Wochen, wobei die effektive Arbeitszeit eine Woche war, da zuerst in das Thema eingeleitet und auf eventuelle Herausforderung bestmöglich vorbereitet wurde.
Im Rahmen des Projektes hat sich so eine Projektgruppe, bestehend aus Nikita Lornik, Julian Fürtig und Tim Müller, gebildet, welche sich für die Konstruktion eines Memory-Roboters entschied. Die klare Zielsetzung lag dabei darauf, ein benutzerfreundliches Interface zu schaffen, mit dem es jeden möglich sein sollte mit ’Emi.mem’ zu interagieren. Außerdem sollte der Roboter logisch und fehlerfrei agieren können. Die Spielidee von Memory ist wahrscheinlich allgemein bekannt, jedoch wird im Folgenden noch einmal das etwas abgewandelte Spielkonzept erläutert. Der Roboter soll die Plättchen einmal einscannen und sich die Farben mit ihren Positionen merken. Im Folgenden werden die Plättchen umgedreht und der Roboter bzw. der Spieler (je nach Spielmodus) drehen Plättchen um. Im Fall des Solo-Modus, in dem der Roboter allein mustergültig Paare findet und zuordnet, wird er alle Paare der Reihe nach aufzeigen. Beim Spielen im Spieler-Modus soll der Roboter Plättchen ”vergessen”, so dass ein echter Spieler auch eine Chance auf den Sieg hat. Der Rest verläuft wie beim normalen Memoryspiel. Zur Entwicklung eines Konzepts wurden verschiedene Ideen in Betracht gezogen, welche folgend erläutert werden.

DOI: 10.24352/UB.OVGU-2020-032

II. Vorbetrachtungen

Zu einem funktionierenden Roboter gehören mehrer Kompenten. Zum einen natürlich das Design bzw. die Kontruktion desgleichen, und zum anderen das äußere Umfeld, Zubehör, sowie die Programmierung und dazugehörige Methoden. Diesbezüglich sind einige Grundideen aufgekommen.

A. Material

Als Material wurde das Lego-Mindstorms-Set zur Verfügung gestellt. Dieses beinhaltet einen NXT-Controller, der als Rechner fungiert, Motoren, Tast-, Farb-, Helligkeits-, Ultraschall-, und Tonsensoren, sowie diverse Legosteine, aus denen der Roboter gebaut werden soll. Es durften jedoch auch weitere Hilfsmittel, wie z.B. eine Webcam verbaut werden. Es gibt aber auch Einschränkungen. So darf zum Beispiel nur ein NXT-Controller verwendet werden, welcher wiederum nur 3 Motoren und 4 Sensoren ansteuern kann. Zusammengefasst sind durch die große Vielfalt an Sensoren und den immerhin 3 Motoren eine relativ breite Varietät an Möglichkeiten, einen Roboter zu entwickeln gegeben.

B. Struktur - Hallenkran

Als Grundstruktur für ’Emi.mem’ wurde ein Hallenkran als Vorbild genommen.

Die Funktionsweise eines solchen ist effektiv wie simpel. Er besitzt zwei bewegliche Lager die senkrecht aufeinander stehen und dem Kran so ermöglichen, die komplette Horizontalbewegung abzudecken. Er kann dadurch nach links/rechts und nach vorne/hinten fahren. Des Weiteren besitzt ein Hallenkran, wie in Abbildung 1 abgebildet, einen herabfahrbaren Haken. So kann dieser mit einfachen Mitteln jeden Punkt in einem 3-dimensionalen Koordinatensystem anfahren. Darüber hinaus ist es spezifisch auf das Memoryspiel bezogen vorteilhaft, da dieses symmetrisch angeordnet ist und der Kran so nicht schräg fahren muss, was die Bedienung um so simpler macht. Jedoch bringt dieses Design auch Nachteile mit sich, da es dem Kran nur schwer möglich ist Kärtchen umzudrehen, weil alle drei, für dieses Projekt verfügbaren, Motoren bereits ausgenutzt waren. Außerdem würde ein solches Gerüst den freien Zugang zum Memoryfeld blockieren, weshalb diese Idee insgesamt noch einige Verbesserungen benötigte.

C. Das ’optimale’ Memoryfeld

Unabhängig von der Ausführung des Designs des Roboters muss auch das Design des Memoryfeldes bestmöglich gewählt werden. Da der Roboter an das Prinzip des normalen Memorys angelehnt sein soll, musste das Spielfeld auch dementsprechend nicht wirklich verändert werden. Damit es dem Zuschauer

Abbildung 1. Modell eines Hallenkrans [1]

nicht zu langweilig wird, und der Roboter auch eine Weile beschäftigt ist, wurde die Größe des Spielfeldes auf 3×4 bzw. 4×4 festgelegt. Außerdem ist auch eine Anzahl von 6 bzw. 8 zu findenden Paaren eine angemessene Menge. Die Wahl der Markierung der Karten war keine Herausforderung, da es durch die vorgegeben Farbsensoren nicht viel Auswahl gab, außer die Karten verschiedenfarbig zu gestalten. Da die Sensoren nur 6 Farben als Wortlaut ausgeben können (rot, blau, gelb, grün, schwarz, weiß), wurde sich letztendlich dazu entschieden zwölf Kärtchen und dementsprechend 6 Paare einzubauen. Die logische Schlussfolgerung, war die Verwendung des 3×4 Memoryfeldes.

D. Programmentwicklung

Die software-technische Umsetzung dieses Projektes sollte in MATLAB stattfinden. Die Programmiersprache ist vergleichbar mit C, jedoch etwas leichter verständlich und vor allem geeigneter um Grafiken auszugeben oder ein Graphical User Interface, auch kurz GUI, zu gestalten.
Die Grundidee für die Programmierung ist, den Code möglichst einfach und verständlich zu schreiben. 'Emi.mem' sollte zwei Spielmodi durchführen können. Im Solo-Modus war generell geplant, dass er mit dem Scannen anfangen und die Farben entsprechend in einer Matrix speichern, diese später wieder auslesen, Paare finden, diese zuordenen und aufzeigen können soll. Die Verwirklichung des Spieler-Modus wurde sich so vorgestellt, dass der Roboter beim Auslesen der Farben zu einer gewissen Wahrscheinlichkeit Felder auslässt und diese so "vergessen"werden.
Ziel war außerdem die Entwicklung eines übersichtlichen und verständlichen GUI, welches dem Spieler die Spielregeln aufzeigt und eine leichte Bedienung ermöglicht.
Mit diesen Grundideen als Voraussetzungen, wurde das Projekt umgesetzt.

III. Umsetzung

In diesem Abschnitt wird auf den Zielpunkt des Projektes und auf Zwischenschritte und -überlegungen desselbigen eingegangen.

A. Aufbau des Roboters

Wie bereits erwähnt, wird der Hallenkran als Vorbild für das Design unseres Roboters genommen. In Abbildung 3 lässt sich erkennen, dass 'Emi.mem' aus zwei Ebenen besteht.

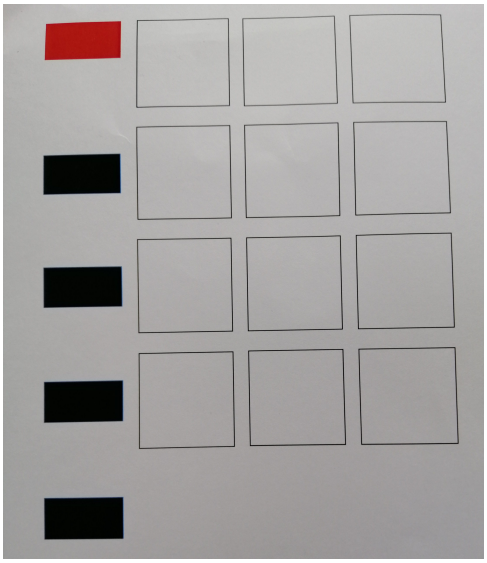


Abbildung 2. Design des Memoryfeldes

Die untere Ebene des Roboters ist auf vier Gummireifen gelagert, welche ihn horizontal nach links und rechts entlang der in Abbildung 2 erkennbaren Rot-Schwarz-Makierung fahren lassen. Dabei sind die Räder als Allradantrieb geschalten, um zu verhindern, dass ein Rad hinterherhinkt und der Roboter so zur Seite abdriftet. Dieser Punkt hatte zunächst einige Schwierigkeiten bereitet, da zu diesem Zeitpunkt kein Motor für eine Lenkung zu Verfügung stand. Durch den neuen Antrieb konnte dieses Problem jedoch relativ simpel gelöst werden.
Auf der zweiten Ebene ist der feste Arm mit Hilfe von 4 Zahnrädern aufgelagert. Dieser lässt sich so auf dieser Ebene horizontal nach vorne und hinten bewegen. Hier wurde sich für Zahnräder entschieden, um den Roboter auf Schienen fahren lassen zu können, damit so, wie bereits beschrieben, das abdriften verhindert werden kann. Am Ende des Armes befindet sich der zu bewegende Farbsensor.
Zu Beginn des Projektes, wurde sich dazu entschieden, diesen Arm beweglich anzubringen, sodass 'Emi.mem' ähnlich wie ein Hallenkran, auch eine vertikal Bewegung durchführen kann. Jedoch hatte dieser keine wirkliche Daseinsberechtigung und hat nur zu weiteren Problemen geführt, weshalb zum Abschluss des Projektes, entschieden wurde, diesen zu demontieren. Das führte wiederum dazu, dass nun wieder einen Motor frei zur Verfügung stand. Da jedoch der Arm bei dem Befahren der zweiten Ebene noch sehr locker gelagert war, wurde der nun übrige Motor einfach als Gegengewicht verwendet.
In Abbildung 3 kann noch ein weiterer Farbsensor auf der linken Seite, einen Tastsensor zu 'Emi.mem's' rechten und den NXT-Controller, an dem der Tastsensor befestigt ist, erkannt werden. Der zweite Farbsensor auf der linken Seite des Roboters, ist dafür zuständig, die Markierungen in Abbildung 2 zu erkennen. Dies erklärt auch, warum diese versetzt zum Memoryfeld angebracht sind. Der Tastsensor war nur für die Präsentation des Solo-Modus von Bedeutung, um den Roboter nach jedem gefundenen Paar anhalten zu lassen und wieder starten zu können. Der NXT-Controller ist an der Seite des Roboters angebracht, um das Gleichgewicht des Konstrukts

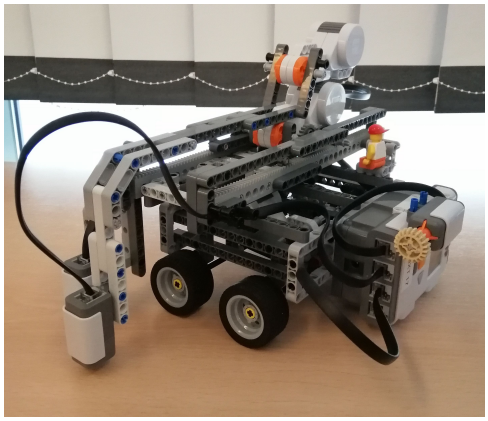


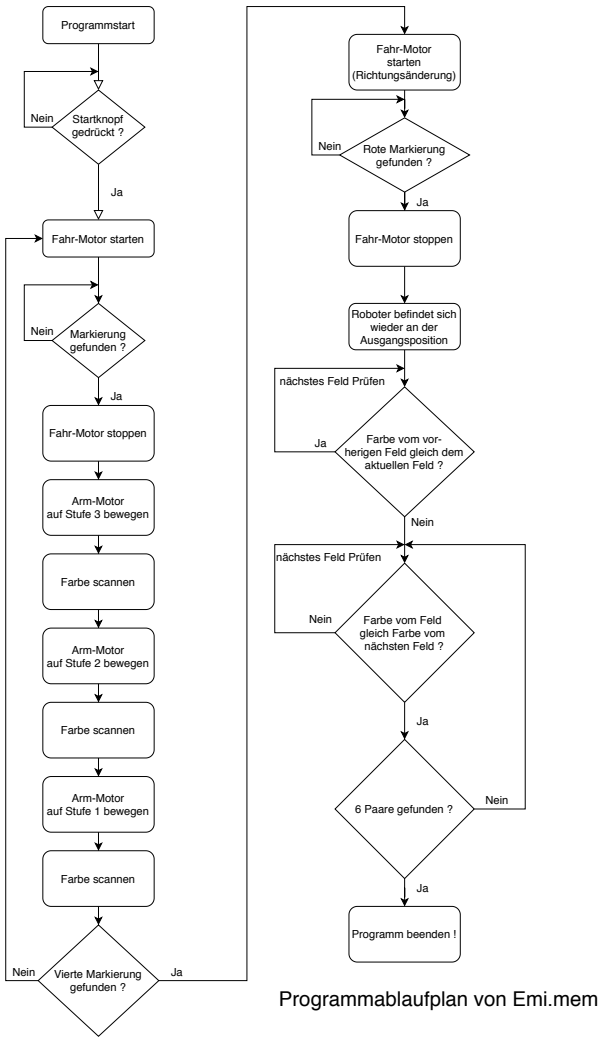
Abbildung 3. ’Emi.mem’

nicht zu sehr zu gefährden. Die Funktion des Controllers ist es, die eingelesenen Daten zu sammeln und den Austausch mit dem Computer zu tätigen.

B. Programmablauf

Der Roboter ist, wie bereits erwähnt, auf 2 Spielmodi ausgelegt, wobei die ersten beiden Schritte komplementär ablaufen.
Zunächst steht ’Emi.mem’ auf dem roten Feld, welches als Startpunkt fungiert. Dann fährt er den ersten schwarzen Balken an und scannt die 3 Felder der entsprechenden Reihe ein. Dies wiederholt sich solange, bis er den vierten schwarzen Balken angefahren hat. Im Folgenden fährt er bis zum roten Startpunkt zurück. Damit ist der Scanvorgang beendet und die Farben werden alle in einer Matrix gespeichert.

Nun beginnen sich die beiden Spielmopdi zu unterscheiden. Im Solo-Modus liest der Roboter die Daten der Matrix einzeln aus und sucht nach Paaren. Sobald er ein Paar gefunden hat, werden beide Felder angefahren und durch einen Signalton aufgezeigt. Im Anschluss wird noch eine Sound-Datei mit der entsprechenden Farbe abgespielt. Dies wird solange wiederholt bis 6 Paare gefunden wurden. Danach begibt sich der Roboter zum Startpunkt zurück und ist wieder einsatz- bereit.
Im Spieler-Modus ist nach Emi.mem’s Scanvorgang zunächst der Spieler am Zug. Nachdem die Karten umgedreht wurden, darf dieser nun versuchen Paare zu finden. Findet er kein Paar mehr, müssen die bereits gefundenen Paare, aus Sicht der Schwarz-Rot-Markierung, in das GUI übertragen werden. So weiß der Roboter, welche Paare noch zu finden sind, ohne nochmal das gesamte Feld scannen zu müssen. Nun geht ’Emi.mem’, wie bereits im Solo-Modus, alle übrigen Felder der Reihe nach durch und überprüft auf Paare. Jedoch findet die Überprüfung nur unter einer gewissen Wahrscheinlichkeit statt. Das bedeutet, dass manche Felder beim Überprüfen der Matrix ausgelassen werden und der Roboter diese sozusagen ”vergisst”. Falls nach diesem Schritt noch nicht alle Felder gefunden wurden, ist der Spieler wieder am Zug und der Vorgang wiederholt sich.
In Abbildung 4 ist der, so eben vorgestellte, Solo-Modus noch einmal veranschaulich dargestellt.



Programmablaufplan von Emi.mem

Abbildung 4. PAP des Solo-Modus

C. GUI - Graphical User Interface

Zu guter Letzt wird noch auf den Aufbau und die Funktionsweise des entworfenen GUI eingegangen.

Es ist erkennbar, dass sich im oberen Bereich des Bildschirmes (Abbildung 5) ein Drop-Down-Menü befindet, in welchem der zu benutzende Spielmodus auszuwählen ist. Etwas darunter befindet sich ein großer, weißer Kasten. Nach der Auswahl des Spielmodus werden hier die Spielregeln und die Funktionsweise des GUI aufgelistet. Nun beginnt der eigentlich spannende Teil. Wird der Solo-Modus ausgewählt, so muss nur auf ’Spiel starten...’ gedrückt werden, und ’Emi.mem’ beginnt seinen Zug. Ist jedoch der Spieler-Modus ausgewählt, so muss zunächst der Knopf ’Emi.mem ist am Zug’ betätigt werden. Der Roboter startet nun das Scannen des Spielfeldes und hält dann wieder an. Nachdem der Spieler nun seinen Zug erledigt hat, muss er gefundene Pärchen in die Matrix, über dem vorher gedrückten Knopf, eintragen. Dies muss aus der Sichtweise des Roboters heraus geschehen. Danach muss ’Übernehmen’ geklickt werden um die Änderungen zu speichern. Mit einem erneuten Klick auf ’Emi.mem ist am Zug’ startet der Roboter seinen Durchlauf.

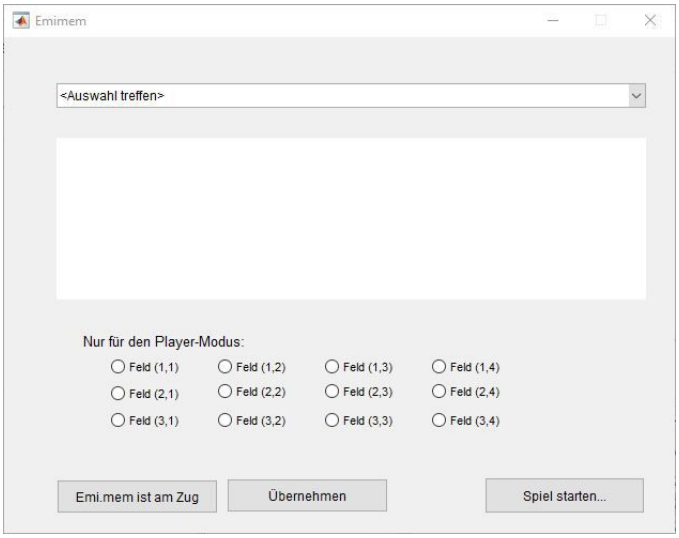


Abbildung 5. Graphical User Interface

Nun geht das Spiel, wie in Abschnitt B. beschrieben, weiter.

IV. Ergebnisdiskussion

Letztendlich ist der Roboter voll funktionsfähig und erfüllt seinen Zweck. Trotz kleinerer Hindernisse, wie einigen Programmierschwierigkeiten und der allgemeinen Stabilität an manchen Stellen, war es möglich den Roboter zu entwerfen und die zu Beginn konstruierte Idee in die Tat umzusetzen. Jedoch gibt es trotzdem kleinere Mängel, die bis jetzt noch nicht komplett behoben werden konnten.
Ein solcher ist zum Beispiel der Farbsensor. Dieser funktioniert unter bestimmten Lichtverhältnissen nur bedingt gut, worauf leider nur relativ wenig Einfluss genommen werden konnte. Als einzige Gegenmaßnahme dafür wurden die Farbsensoren so nah wie möglich an der Oberfläche angebracht, um die optimale Farbaufnahme zu ermöglichen. Da der Roboter jedoch essentiell auf die beiden Farbsensoren zählt, kann das gesamte Programm, durch einen Fehler beim Scannen der Farben, nicht mehr richtig funktionieren.
Ein weiteres Problem ist das Spurhalten des Roboters. Es konnte zwar die obere Ebene konstant zum geradeaus fahren gebracht werden, jedoch kann es beim längeren Benutzen oder ungenauem Ausrichten der unteren Räder, schnell zum abdriften kommen. Trotz einiger Versuche, ist es nicht gelungen hier eine geeignete Schiene zu entwerfen, da diese meist mit dem zweiten Farbsensor kollidiert ist. Jedoch kann das Problem manuell, durch ein leichtes Drehen des Memoryfeldes, behoben werden.
Ein letztes Problem ist die lange Scanzeit, die man mit einer Webcam hätte verkürzen können. Der Grund für die Entscheidung der Verwendung eines Farbsensors statt Webcam, ist größtenteils dem Projektnamen geschuldet. Da sich an den im Set enthaltenen Materialien orientiert wurde lag hier die Idee, die von Lego gestellten Farbsensoren zu verwenden, nicht allzu fern, auch wenn dieser Weg etwas unpraktischer und umständlicher ist.

V. Zusammenfassung und Fazit

Insgesamt ist die Entwicklung des Memoryroboters sehr zufriedenstellend abgelaufen. Das Projekt bereitete viel Freude, insbesondere durch die Möglichkeit eigene Ideen realisieren zu können. Das Projekt wurde innerhalb der gegebenen Zeit umgesetzt, und der Bau- und Programmiervorgang konnten, ohne große Vorkenntnisse des Teams zu diesem Thema, erfolgreich zum Abschluss gebracht werden.
Hätte mehr Zeit zur Verfügung gestanden, so hätten auch noch weitere Ideen in den Roboter eingebracht werden können. Es könnte beispielsweise, der dritten Motor für eine Lenkung verwendet werden, die das Abdriften des Roboters verhindert. Des Weiteren wäre auch eine Implementierung verschiedener Schwierigkeitsgrade für den Spieler-Modus möglich. Außerdem könnte ein ganz neuer Spielmodus hinzugefügt werden, in dem, wie beim echten Memory, gespielt wird, ohne die Plättchen vorher zu Scannen und sich der Roboter per Trial-and-Error die Pärchen sucht. Hätte man mehr Motoren und mehr Anschlüsse, also dementsprechend mehr NXT-Controller, könnte man auch eine Apparatur bauen, die es dem Roboter ermöglicht, die Plättchen selbstständig umzudrehen.
Wie man sehen kann, sind die Grenzen eines Memoryroboters unendlich. Man kann das Konzept auch auf andere Spiele übertragen und 'Emi.mem' so zu einem 'Universal-Gegner' in jeglichem Spiel verwandeln. Jedoch ging es in diesem Projekt nicht darum, eine Maschine zu entwickeln, welche die Anwendung von Robotoren im Allgemeinen neu definiert, sondern nur darum, sowohl den Studenten, als auch jedem der dies liest, zu zeigen, dass die Automatisierung, die zur Zeit in jedem Lebensbereich stattfindet, nichts übermenschliches ist und trotzdem grenzenlose Möglichkeiten offenbart.

Literaturverzeichnis

[1] ABUS - Mehr bewegen: *Hallenkrane/Laufkrane*. https://www.abus-kransysteme.de/krane/laufkrane. Version: März 2020