

Optimierung Hierarchischer Fuzzy-Regler mit Genetischen Algorithmen

Frank Hoffmann und Gerd Pfister

Universität Kiel
Institut für Angewandte Physik
24098 Kiel, Germany
Telefon: 0431-880-3978 , Fax: 0431-880-4608
Email: hoefi@ang-physik.uni-kiel.de

Zusammenfassung Ein Verfahren zur Konstruktion von hierarchischen Fuzzy-Reglern mit Genetischen Algorithmen wird vorgestellt. Am Beispiel der Steuerung eines autonomen Fahrzeugs wird eine Anwendung der Methode demonstriert. Das Fahrzeug hat die Aufgabe ein vorgegebenes Ziel zu erreichen und dabei eine Kollision mit Wänden und Hindernissen zu vermeiden.

1 Einführung

Der Entwurf der linguistischen Variablen und der Regelbasis eines Fuzzy-Systems erfordert Expertenwissen. Dieses Wissen erhält man durch Befragung oder Beobachtung eines menschlichen Bedieners des zu regelnden dynamischen Systems. Neben Adaptiven Fuzzy-Systemen und Neuro Fuzzy-Systemen [1] wurden Genetische Algorithmen (GA) zur automatischen Generierung von Fuzzy-Reglern vorgeschlagen. GA sind Optimierungsverfahren, die mit Operatoren, die aus der natürlichen Evolution stammen arbeiten [2]. Zur Optimierung von Fuzzy-Reglern konnten GA erfolgreich eingesetzt werden [3]. Ein Ansatz zur Verbesserung eines existierenden Reglers ist die Modifikation der Zugehörigkeitsfunktionen der Fuzzy Mengen [4]. Eine andere Methode optimiert die Regelbasis [5] mit Hilfe eines GA. Im folgenden wird ein Verfahren zur Konstruktion von hierarchischen Fuzzy-Reglern mit einem GA vorgestellt. Der Aufbau des hierarchischen Fuzzy-Reglers mit versteckten Fuzzy-Variablen wird im Kapitel 2 vorgestellt. Kapitel 3 beschreibt den GA der zur Optimierung der Regelbasis benutzt wird. Zusätzlich zu den bekannten Operationen Selektion, Crossover und Mutation, wird der GA um einen Neuordnungsoperator erweitert, der dafür sorgt, daß nach dem Crossover wieder eine "sinnvolle" Regelbasis entsteht. In Kapitel 4 wird die Methode zum Entwurf eines Fuzzy-Reglers für ein autonomes Fahrzeug verwendet.

2 Aufbau des hierarchischen Fuzzy-Reglers

Die Anzahl der Regeln eines Fuzzy-Systems wächst überproportional mit der Anzahl der linguistischen Variablen und der zugehörigen linguistischen Terme.

Für eine vollständige Regelbasis mit linguistischen Eingangsvariablen $\{X_i | i = 1, \dots, n\}$ mit Termen $\{A_{ij} | j = 1, \dots, q_i\}$ und linguistischen Ausgangsvariablen $\{Y_i | i = 1, \dots, l\}$ mit Termen $\{B_{ij} | j = 1, \dots, p_i\}$ benötigt man N Regeln mit

$$N = \prod_{i=1}^n q_i . \quad (1)$$

Die Regeln haben die Form:

if $X_1 = A_1$ **and** ... **and** $X_n = A_n$ **then** $Y_1 = B_1$ **and** ... **and** $Y_l = B_l$

mit $A_i \in \{A_{ij}\}, B_i \in \{B_{ij}\}$.

Die große Anzahl von Regeln erschwert den Entwurf einer Regelbasis, da vom Experten für jede der N verschiedenen Prämissen eine Kombination der linguistischen Ausgangsvariablen angegeben werden muß. Auf einen Teil der Regelbasis kann verzichtet werden, wenn gewährleistet ist, daß bestimmte Kombinationen von Eingangsgrößen im dynamischen System nicht auftreten. Ein solcher Beweis erfordert ein analytisches Modell des zu regelnden Systems.

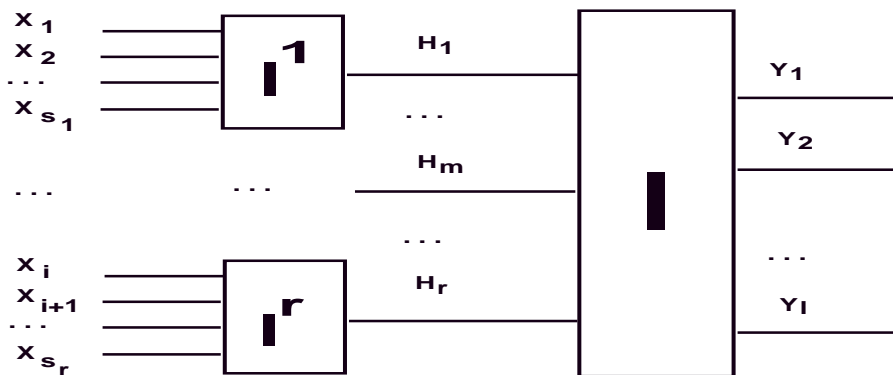


Abbildung1. Aufbau des hierarchischen Fuzzy-Reglers

Ein hierarchischer Aufbau der Regelbasis erlaubt es, die Anzahl der Regeln zu reduzieren. Dazu führt man in Analogie zu den versteckten Neuronen in Neuronalen Netzwerken versteckte Fuzzy-Variablen ein. Der Aufbau des hierarchischen Fuzzy-Reglers ist in Abb. 1 dargestellt.

Zunächst werden die n linguistischen Eingangsvariablen X_i in disjunkte Mengen $\{Q_m | m = 1, \dots, r\}$ mit jeweils s_k Eingangsvariablen unterteilt. Jeder Menge Q_m wird eine vollständige Regelbasis I^m mit Regeln R^m der Form

if $X_1 = A_1$ **and** ... **and** $X_{s_m} = A_{s_m}$ **then** $H_m = C_m$

zugeordnet. Die H_m im Konklusionsteil der Regeln R^m sind die versteckten Fuzzy-Variablen, zu denen kein scharfer Wert einer reellen Größe existiert. Die $C_m \in \{C_{mj} | j = 1, \dots, p_m\}$ entsprechen den Termen bei herkömmlichen Ausgangs-Fuzzy-Variablen. Ihren Zugehörigkeitswert $\mu(C_{mj})$ erhält man durch Inferenz der Regelbasis I^m . Der Zugehörigkeitswert $\mu(B_{ij})$ der Ausgangsvariablen Y_i wird durch Inferenz der Regelbasis I bestimmt, mit Regeln der Form

if $H_1 = C_1$ **and** ... **and** $H_r = C_r$ **then** $Y_1 = B_1$ **and** ... **and** $Y_l = B_l$.

Die so erhaltenen Fuzzy Mengen B_{ij} werden durch Defuzzifizierung in scharfe Ausgangswerte y_i abgebildet. Die Zuordnung verschiedener Prämissen zum gleichen versteckten Fuzzy-Term C_{mj} entspricht einer Fuzzy-ODER Verknüpfung dieser Prämissen. Allen Prämissen der Regelbasis I_m , die den gleichen versteckten Fuzzy-Term C_{mj} als Konklusion besitzen, wird in der nachfolgenden Regelbasis I die gleiche Konklusion auf die Ausgangsvariablen B_i zugeordnet.

Seien P_i^m die Prämissen die zum gleichen versteckten Fuzzy-Term H_m gehören. Die resultierende Gesamtregelbasis läßt sich darstellen mit Regeln der Form

if (P_1^1 **or** P_2^1 **or** ...) **and** (P_1^2 **or** P_2^2 **or** ...) **and** ...
...and (P_1^r **or** P_2^r **or** ...) **then** $Y_1 = B_1$ **and** ... **and** $Y_l = B_l$.

Durch die Zuordnung verschiedener Prämissen zum gleichen versteckten Fuzzy-Term wird die Anzahl der Regeln N deutlich reduziert.

$$N = \sum_m N_{I_m} + N_I \quad (2)$$

Dabei sind N_{I_m} , N_I die Anzahl der Regeln der Basen I^m , I , die man aus (1) erhält, mit einer kleineren Anzahl von Eingangsvariablen n im Vergleich zu einem nichthierarchischen Fuzzy-Regler.

3 Genetische Algorithmen

Bei einem GA werden die Parameter des Optimierungsraumes durch einen binären String fester Länge S_i codiert. Der GA arbeitet mit einer Population $P = \{S_1, S_2, \dots, S_M\}$ von Strings, welche sich durch die genetischen Operatoren Selektion, Crossover und Mutation von einer Generation zur nächsten weiterentwickelt. Die Fitnessfunktion bewertet die Güte von Parameterkombinationen im Optimierungsraum, welche durch Strings codiert werden. Die Selektion bewirkt, daß sich Strings mit hoher Fitness mit größerer Wahrscheinlichkeit in die nächste Generation fortpflanzen. Beim Crossover werden aus der Kombination zwei Eltern-Strings neue Nachkommen für die nächste Generation gebildet.

Die Mutation modifiziert einzelne Bits eines Strings und vermeidet dadurch die vorzeitige Konvergenz des GA in lokalen Minima.

Zur Optimierung des oben beschriebenen hierarchischen Fuzzy-Reglers mit GA muß zunächst eine geeignete Codierung gewählt werden. Um die Vollständigkeit des Fuzzy-Reglers zu gewährleisten, muß für jede der möglichen Prämissen eine Konklusion festgelegt werden. Die Konklusionen der Regelbasen I_m und I werden dazu durch binäre Strings codiert. Die Konklusion $H_m = C_m$ der Regel R_m kann einen von p_m möglichen Terme C_{mj} annehmen. Ein gewählter Term C_{mj} wird als ganze Zahl j durch einen Teilstring TS_m der Länge $ld(p_m)$ codiert. Zusätzlich kann jeder Regel ein Gewicht $w_m \in [0, 1]$ zugeordnet werden, welches den Bedeutungsgrad dieser Regel widerspiegelt. Die Gewichte werden linear auf ganze Zahlen mit l Bit Genauigkeit abgebildet, welche an die binären Teilstrings TS_m angehängt werden. Den Teilstring S_m für eine Regelbasis I_m erhält man durch Aneinanderfügen der Teilstrings TS_m . Analog erfolgt die Codierung der Konklusionen $Y_1 = B_1^k$ and $\dots Y_m = B_m^k$ in einem Teilstring TS , wobei für jede Ausgangsvariable Y_i der Ausgangsterms B_{ij} wiederum als ganze Zahl j codiert wird. Die Teilstrings TS bilden zusammengesetzt den Teilstring S für die Codierung der Regelbasis I . Den gesamten String, der einen Fuzzy-Regler codiert, erhält man durch Aneinanderfügen der Teilstrings aller Regelbasen $S_1 S_2 S_3 \dots S_r S$.

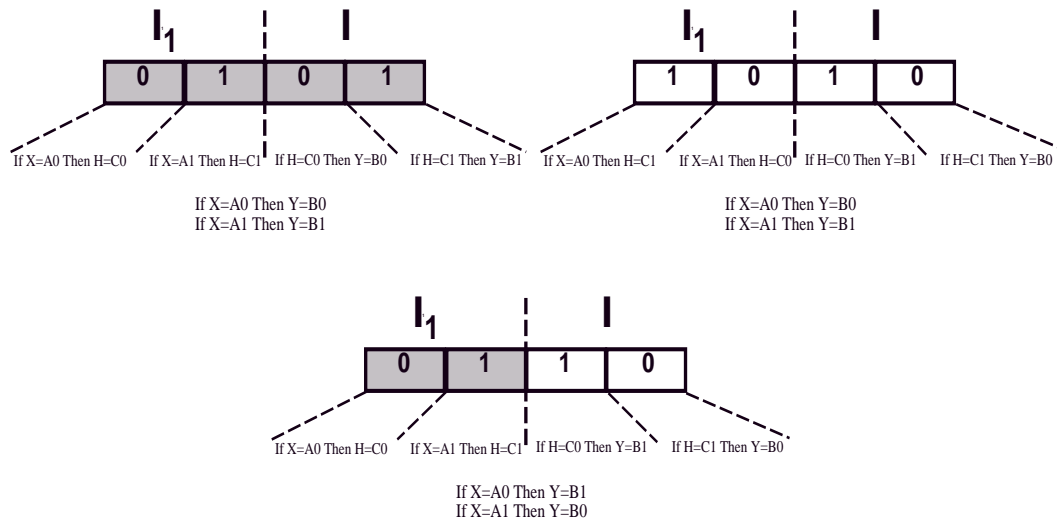


Abbildung 2. Vertauschung der versteckten Fuzzy-Terme beim Crossover

Der hierarchische Fuzzy-Regler ist invariant unter einer Permutation der Indizes j der Terme C_{mj} der versteckten Fuzzy-Variablen H_m , wenn diese Permutation gleichzeitig im Konklusionsteil der Regeln R_m und im Prämissenteil der

Regeln R stattfindet. Beim Crossover entsteht ein neuer Regler durch Kombination zweier Strings der Eltern A und B . Crossover bewirkt eine Aufspaltung der Elternstrings. Ein Nachkomme kann die Regelbasen I^m vom Elternteil A und die Regelbasis I vom Elternteil B erben. Dabei ist nicht gewährleistet, daß die Indizes j der versteckten Fuzzy-Terme C_{mj} in den Regelbasen I^m und I der beiden Eltern in der gleichen Reihenfolge verwendet werden. Dies führt zu falschen Prämissen in der Regelbasis I , da diese nicht mehr mit den neuen Konklusionen der Regelbasen I^m übereinstimmen. Abb.2 verdeutlicht diese Problematik.

Der einfachste nicht triviale hierarchische Fuzzy-Regler besitzt eine Eingangsvariable X mit Termen A_0, A_1 , eine versteckte Fuzzy-Variable H mit Termen C_0, C_1 und eine Ausgangsvariable Y mit Termen B_0, B_1 . Jede der Regeln läßt sich durch ein einzelnes Bit codieren, da der Konklusionsteil einen von zwei möglich Termen enthalten kann. Beide Eltern $[0101], [1010]$ entsprechen trotz unterschiedlicher Codierung dem einfachen nichthierarchischem Fuzzy-Regler:

if $X = A_0$ **then** $Y = B_0$, **if** $X = A_1$ **then** $Y = B_1$

Nach dem Crossover entsteht der Nachkomme $[0110]$ mit den Regeln

if $X = A_0$ **then** $Y = B_1$, **if** $X = A_1$ **then** $Y = B_0$

welcher ein gegenteiliges Regelverhalten aufweist. Um dieses unerwünschte Verhalten beim Crossover zu vermeiden, muß eine Zuordnung der Terme beider Eltern erfolgen, indem die Terme C_{mj} in einer bestimmten festgelegten Reihenfolge angeordnet werden. Zu diesem Zweck benutzt unser GA einen Neuordnungsoperator, damit beim Crossover der Elternstrings wieder "sinnvolle" Regelbasen entstehen. Dieser Neuordnungsoperator für die versteckten Fuzzy-Terme ähnelt dem Anordnen der Regeln zweier Eltern, wie es Cooper [8] vorgeschlagen hat. Dabei werden die Regeln zweier Strings so angeordnet, daß die Schwerpunkte der Eingangsterme so weit wie möglich übereinstimmen.

Der Neuordnungsoperator benötigt für jede Regelbasis I^m eine Abbildung F_m , die jeder Prämisse P_m einen Wert $F_m(P_m)$ zuordnet. Für jeden Term C_{mj} bildet man den Mittelwert von F_m über alle Prämissen P_m , die C_{mj} als Konklusion besitzen.

$$F_{mj} = \langle F_m(P_m) \rangle_{P_m \Rightarrow C_{mj}} \quad (3)$$

Die neue Reihenfolge der Terme C_{mj} ergibt sich aus den reellen Werten F_{mj} . Die Indizes j der Terme C_{mj} werden so sortiert, daß $F_{m1} \leq F_{m2} < \dots < F_{mp_m}$ gilt. Dieser Neuordnungsoperator wird auf alle Strings der Population angewandt. Dies gewährleistet eine einheitliche Interpretation der Fuzzy-Variablen H_m durch die Regelbasis I . Durch Wahl der Abbildung F_m erhalten die versteckten Fuzzy-Variablen H_m eine Bedeutung.

Für die Selektion muß eine Fitnessfunktion gewählt werden, welche die Güte des Fuzzy-Reglers im Hinblick auf das gewünschte Verhalten des dynamischen Systems beschreibt [5]. Falls ein Referenzdatensatz mit geeigneten Steuerungsanweisungen für verschiedene Zustände des Systems existiert, so kann dieser zur

Optimierung des Reglers verwendet werden. Als Fitnessfunktion dient in diesem Fall die Abweichung der Ausgangsgrößen des Fuzzy-Reglers zu denen des Referenzdatensatzes [4].

4 Autonomes Fahrzeug

Für ein autonomes Fahrzeug wurde mit der oben vorgestellten Methode ein Fuzzy-Regler entworfen. Das Fahrzeug soll innerhalb geschlossener Räume zuvor festgelegte Zielpunkte selbständig ansteuern. Im Weg befindliche Hindernisse, wie Wände oder Gegenstände sollen erkannt und umfahren werden. Drei Ultraschallsensoren ermöglichen eine berührungslose Abstandsmessung zu Hindernissen. Die Steuerung des Fahrzeugs erfolgt über zwei unabhängige Schrittmotoren. Die aktuelle Position und Orientierung des Fahrzeugs erhält man durch Mitzählen der Schritte beider Antriebsräder.

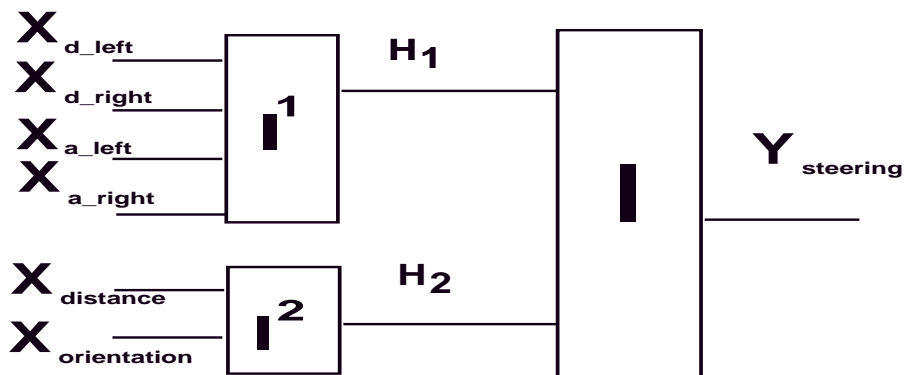


Abbildung3. Aufbau des Fuzzy-Reglers für das Fahrzeug

Der Fuzzy-Regler besitzt als Eingangsgrößen die von den Ultraschallsensoren erhaltenen Abstände zu Hindernissen links und rechts X_{d_left} , X_{d_right} . Jede der Eingangsvariablen X_{d_left} , X_{d_right} enthält drei trapezförmige Fuzzy-Terme *nah*, *mittel*, *weit*, welche die Eingangswerte von 0 ... 3m überdecken. Durch Vergleich des zwischen zwei Messungen zurückgelegten Weges des Fahrzeuges mit der Änderung der Abstände berechnet man die Winkel X_{a_left} , X_{a_right} zu Hindernissen. Die Eingangsgrößen X_{a_left} , X_{a_right} besitzen die Fuzzy-Terme *flach* und *steil*, die angeben, ob sich ein Hindernis eher in Fahrtrichtung oder parallel zur Fahrtrichtung befindet. Die beiden Terme *flach* und *steil* überdecken die Eingangswerte von $0^\circ \dots 90^\circ$.

Weitere Eingangsgrößen sind der Abstand zum Zielpunkt $X_{distance}$ und die relative Orientierung X_{orient} der Fahrzeugachse zum Ziel. Die Eingangsvariable $X_{distance}$ besitzt die beiden Fuzzy-Terme *nah* und *weit*. X_{orient} beinhaltet die fünf Fuzzy-Terme *sehr links*, *links*, *voraus*, *rechts* und *sehr rechts*.

Ausgangsgröße ist der Steuerwinkel $Y_{steering}$, der in die Schrittgeschwindigkeiten v_1 und v_2 der beiden Motoren umgerechnet wird. Diese setzt sich aus fünf Fuzzy-Termen *stark links*, *links*, *geradeaus*, *rechts* und *stark rechts* zusammen.

Als Inferenzmethode wird Min-Max-Inferenz verwendet. Zum Berechnen der scharfen Ausgangsgröße der Variablen $Y_{steering}$ wird Center-of-Gravity zur Defuzzifizierung benutzt. Als Verknüpfungoperator der Prämissen einschließlich der Gewichte wird der Minimumsoperator verwendet.

Die vier Fuzzy-Variablen $X_{d_{left}}$, $X_{d_{right}}$, $X_{a_{left}}$ und $X_{a_{right}}$ werden in der Regelbasis I^1 zusammengefaßt. Die beiden verbleibenden Fuzzy-Variablen $X_{distance}$ und X_{orient} bilden den Eingang der Regelbasis I^2 . Die Inferenz auf die Ausgangsvariable Y_{steuer} erfolgt in der Regelbasis I , die als Eingänge die versteckten Fuzzy-Variablen H_1 und H_2 erhält. Der Aufbau des Reglers ist in Abb. 3 zu sehen. Die Aufgabe der versteckten Fuzzy-Variablen H_1 liegt in der Erkennung von Hindernissen durch Auswertung der Abstandsmessungen der drei Ultraschallsensoren. H_2 repräsentiert Informationen über die Lage des Zielpunktes. Dies entspricht dem Ansatz von Ruspini [7], der durch kontextabhängiges Überblenden der beiden Einzelziele Kollisionsvermeidung und Zielansteuerung zu einer Gesamtstrategie gelangt. Die versteckte Fuzzy-Variable H_1 enthält vier, H_2 enthält fünf Terme. Die Gesamtanzahl der Regeln beträgt 66 im Vergleich zu einem nichthierarchischem Regler, der 360 Regeln benötigt.

Die Abbildung F_1 ergibt sich aus der Differenz der Indizes der Fuzzy-Terme $A_{d_{left}} \in \{ nah, mittel, weit \}$ und $A_{a_{left}} \in \{ flach, steil \}$ zu denen der Fuzzy-Variablen $A_{d_{right}}$, $A_{a_{right}}$. Die Prämisse P

if $X_{d_{left}} = A_{j_1}$ **and** $X_{d_{right}} = A_{j_2}$ **and** $X_{a_{left}} = A_{j_3}$ **and** $X_{a_{right}} = A_{j_4}$

bekommt den Wert $F(P) = (j_1 - j_2) + (j_3 - j_4)$ zugewiesen. Die Terme C_{1j} der versteckten Fuzzy-Variablen H_1 geben an, ob sich ein Hindernis links, vor oder rechts vom Fahrzeug befindet.

Die Abbildung F_2 verwendet nur den Fuzzy-Term $A_{orient} \in \{ sehr links, links, voraus, rechts, sehr rechts \}$. Die Prämisse P

if $X_{orient} = A_{j_1}$ **and** $X_{distance} = A_{j_2}$

bekommt den Wert $F(P) = j_1$ zugewiesen. Die versteckten Fuzzy-Terme C_{2j} geben an, ob sich das Ziel links, vor oder rechts vom Fahrzeug befindet.

Zur Beurteilung der Qualität eines Reglers wurde das Fahrzeug in einer Simulation in verschiedenen Umgebungen getestet. Die Umgebungen unterschieden sich durch die Art und Anzahl der Hindernisse, sowie durch die Lage des Zielpunktes. Es zeigte sich, daß der GA statt der eigentlichen Steuerungsaufgabe die Geometrie der Räume und Hindernisse erlernte. Dies führte dazu, daß der Regler zwar die Aufgabe für die vorgegebenen Testumgebungen löste, jedoch in nicht zuvor präsentierten Umgebungen scheiterte. Um dies zu vermeiden, wurden Start- und Zielpunkte, sowie die Lage und Größe der Hindernisse in jeder Generation geändert.

Ein Regler erhielt dann eine hohe Fitness, wenn es dem Fahrzeug gelang ohne Kollision mit einem der Hindernisse, den Zielpunkt möglichst genau und auf kurzem Wege zu erreichen. Die Fitnessfunktion in unserer Optimierungsaufgabe beschreibt die Qualität des gesamten Fuzzy-Reglers. Diese Fitnessfunktion kann erst nach Erreichen des Zielpunktes, oder nach einer Kollision ermittelt werden. Die Beurteilung der Güte einzelner Regeln ist nicht möglich, da nicht bekannt ist, welche der zurückliegenden Steuerungsanweisungen für den Erfolg oder Mißerfolg verantwortlich waren. Diese als zeitliches "Credit Assignment Problem" bekannte Schwierigkeit taucht immer dann auf, wenn mehrere Entscheidungen zu aufeinanderfolgenden Zeitpunkten zu einer Gesamtstrategie gehören.

Von einer maximal erreichbaren Fitness F_{max} wurden Strafen für eine Kollision P_{col} , für den Abstand zum Zielpunkt P_{target} und den zurückgelegten Weg $P_{distance}$ abgezogen. Jeder Regler wurde in zwölf Umgebungen getestet. Die Gesamtfitness F ergab sich aus der Summe der Fitnessfunktion der einzelnen Räume:

$$F = \sum_{\text{Räume}} F_{max} - P_{col} - P_{target} - P_{distance} \quad (4)$$

Abgesehen vom Neuordnungsoperator wurde ein Standard GA mit Crossover, Mutation und skaliertes Fitnessfunktion für die Selektion benutzt. Bei einer Populationsgröße von 50 Strings fand der GA nach 200 Generationen einen Fuzzy-Regler, welcher die Steuerungsaufgabe bewältigt.

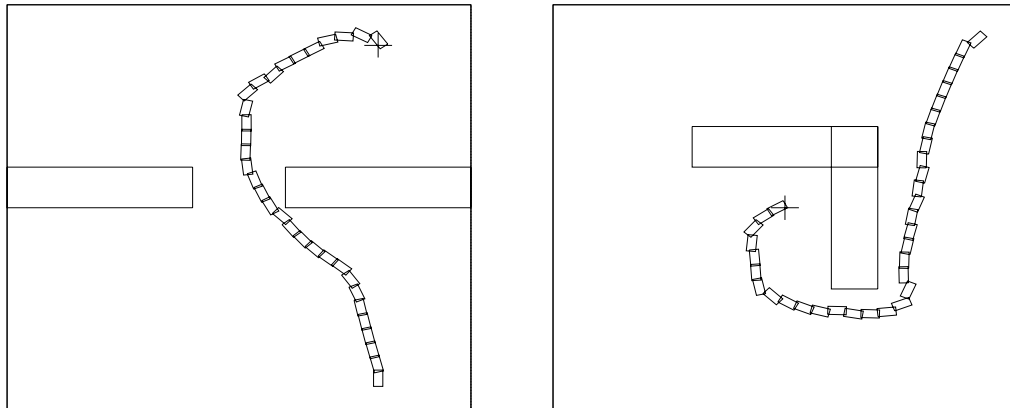


Abbildung4. Beispiel für zwei simulierte Fahrten des autonomen Fahrzeugs

Abb. 4 zeigt zwei simulierte Fahrten des autonomen Fahrzeugs. Die anzusteuenden Zielpunkte sind mit einem Kreuz markiert.

5 Zusammenfassung

Ein Fuzzy-Regler mit hierarchischem Aufbau benötigt im Vergleich zu einem nichthierarchischem Regler eine geringere Anzahl von Regeln. Versteckte Fuzzy-Variablen faßen mehrere Prämissen zu einer Gruppe zusammen, wodurch die Regelbasis in kleinere Teile unterteilt werden kann. In dieser Arbeit wurde ein Entwurfsverfahren zum Erstellen hierarchischer Fuzzy-Regler mit Hilfe Genetischer Algorithmen vorgestellt. Es wurde ein Neuordnungsoperator eingeführt, der gewährleistet, daß die versteckten Fuzzy-Terme von allen Strings der Population in der gleichen Weise interpretiert werden. Die Anwendbarkeit der Methode wurde am Beispiel der Steuerung eines autonomen Fahrzeugs demonstriert.

References

1. Kais Brahim, "Neuro-Fuzzy Inferenz-Systeme", *3. Dortmunder Fuzzy-Tage*, Springer-Verlag, 1993, Seite 176-186
2. D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Reading Massachusetts: Addison-Wesley, 1989
3. C. L. Karr, L. M. Freeman, D. L. Meredith, "Improved fuzzy process control of spacecraft autonomous rendezvous using a genetic algorithm", *SPIE Intelligent Control and Adaptive Systems*, vol. 1196, pp. 274-288, 1989
4. Hartmut Surmann, Andreas Kanstein, Karl Gosser, "Self-Organizing and Genetic Algorithms for an Automatic Design of Fuzzy Control and Decision Systems", *EUFIT 93 First European Congress on Fuzzy and Intelligent Technologies*, Aachen 1993, Vol. 2, pp. 1097-1104
5. K. Kropp, U. G. Baitinger, "Optimization of Fuzzy Logic Controller Inference Rules Using a Genetic Algorithm", *EUFIT 93 First European Congress on Fuzzy and Intelligent Technologies*, Aachen 1993, Vol. 2, pp. 1090-1096
6. Marco Dorigo, Uwe Schnepf, "Genetics-Based Machine Learning and Behaviour-Based Robotics: A New Synthesis", *IEEE Transactions on Systems, Man, and Cybernetics*, January/February 1993, Vol. 23, No. 1, pp. 141-153
7. Alessandro Saffiotto, Enrique H. Ruspini, Kurt Konolige, "A Fuzzy Controller for Flakey an Autonomous Mobile Robot", *3. Dortmunder Fuzzy-Tage*, Springer-Verlag, 1993, pp. 3-12
8. Mark G. Cooper, Jaques J. Vidal, "Genetic Design of Fuzzy Controllers", *Second International Conference on Fuzzy Theory and Technology*, Durham, NC, October 1993