

Evolutionäre Algorithmen

Prof. Dr. R. Dillmann

Prof. Dr.-Ing. J. Marius Zöllner



Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft



Universität Karlsruhe (TH)
Forschungsuniversität • gegründet 1825

Evolutionäre Algorithmen

Biologisches Evolutionsmodell nach Darwin:

Selektion = Treibende Kraft der Evolution

„Man kann die natürliche Selektion genauso gut in Formeln packen, wie es mit natürlichen Neuronalen Netzen geht.“

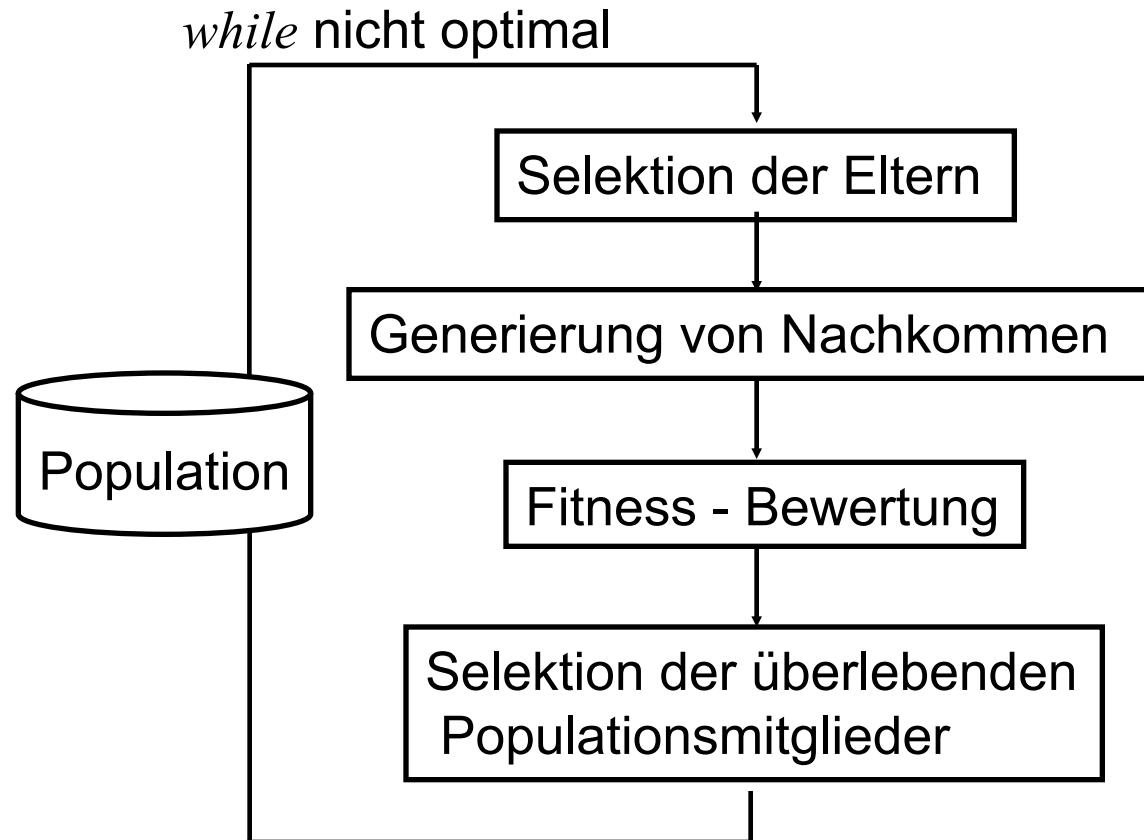
Technisch: Evolution als Optimierung komplexer, künstlicher Systeme

durch Zufall und Selektion → Erzeugung hochkomplexer und
an die Arbeits- bzw.
„Lebensumgebung“
hervorragend angepasster
Systeme

Wurzeln: Rechenberg ('60), Holland('75), Goldberg('89)

- Individuum: \Leftarrow eine mögliche Lösung, Hypothese
- Population und Generation \Leftarrow Lösungs- bzw. Hypothesenmenge
- Erzeugen von Nachkommen \Leftarrow Generierung neuer Hypothesen
 - Methoden z.B.: Rekombination
 - Mutation
- Veränderter Nachfolger, Kind, Nachkomme \Leftarrow neue Hypothese
- Fitness (-funktion) \Leftarrow Hypothesengüte, zu optimierendes Kriterium
- „Selektion der Besten“ \Leftarrow Auswahl der Hypothesen, welche die beste Problemlösung erzeugen

Der Grundalgorithmus



- Wissen wird meistens strukturiert repräsentiert
- Kodieren durch Gene
 - k-Alphabet ($k=2$: Binärcodierung) \Leftarrow Genetische Algorithmen
 - Reelle Zahlen, Vektoren \Leftarrow Evolutionäre Strategien
 - baumartige Strukturen \Leftarrow Genetisches Programmieren

Beispiel: wenn $x=T$ und $y=F$ dann $z=T$;

10 01 1

↓ ↓ ↓

Binärcodierung ($k=2$) : 1001|1110|1111|1010|1000|1010

- Wie viel von dieser Strukturinformation soll genutzt werden?
 - kein Einsatz, ausschließliches Anwenden der Algorithmen z.B. auf den Binärsequenzen
 - volle Ausnutzung der Strukturinformation \Rightarrow Spezielles Anpassen der Optimierungsalgorithmen

Generierung von Nachkommen

- Erfolgt durch s.g. genetische Operatoren
- Zwei Strategien:

Exploration \leftrightarrow Exploitation

Erforschen des Hypothesenraumes \leftrightarrow Lokale Optimierung

Vergleich:

- Je stärker und zufälliger Änderungen sind, um so geringer ist die Wahrscheinlichkeit, bessere Nachkommen zu erzeugen
- Bei lokalen Verbesserungsmethoden ist die Gefahr der lokalen Minima gegeben
- Explorationsgrad muss gemäß der aktuellen Fitness der Generation ausgewählt werden (z.B.: anfangs hoch dann fallend)

Der Nachkomme stammt von einem Elternteil ab

- Mutation einzelner Gene

Elternteil:

z.B. Bitinversion

1 0 1 0 1 1 0

Nachkomme:

1 0 1 1 1 1 0

- Konzepte:

- Alle Bits einer Sequenz werden unabhängig voneinander mit einer bestimmten Wahrscheinlichkeit invertiert
- Für eine bestimmte (oder zufällige) Anzahl von Bits werden die Indizes zufällig ausgewählt
- Stochastisch bei kontinuierlicher Repräsentation :

$$x_i := x_i + z,$$

wobei z eine Zufallsvariable
z.B.: nach $N(0, \sigma)$ Normalverteilung
um 0 mit Varianz σ

Mutation

- Mutationsoperator bei Sequenzen
 - Herausnehmen einer Teilsequenz und Einfügen an anderer Stelle
 - Invertiertes Einfügen der Teilsequenz

Sequenz Inversion

Elternteil:

 $x_1 \dots x_i \textcolor{blue}{x_{i+1}} \dots \textcolor{blue}{x_j} x_{j+1} \dots x_n$

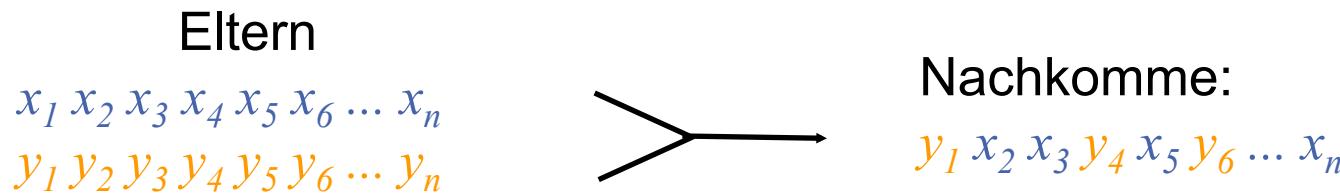
Nachkomme:

 $x_1 \dots x_i \textcolor{blue}{x_j} \dots \textcolor{blue}{x_{i+1}} x_{j+1} \dots x_n$

- Spezielle Mutationsoperatoren → anwendungsspezifisch

Eigenschaften von zwei oder mehreren Eltern sollen gemischt werden

- Diskrete Rekombination



- Intermediäre Rekombination (kontinuierliche Repräsentation)

Sei $Elternteil_1 := x$ und $Elternteil_2 := y$, dann ist Nachkomme z definiert durch:

$$z_i := (\underline{x}_i + \underline{y}_i)/2$$

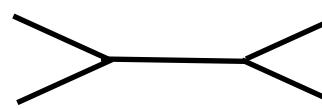
Rekombination

- Crossover – aus 2 Eltern → 2 Nachkommen
 - Single-point crossover

$$\begin{array}{ccccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & \dots & x_n \\ y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & \dots & y_n \end{array}$$


$$\begin{array}{ccccccccc} y_1 & y_2 & y_3 & y_4 & x_5 & x_6 & \dots & x_n \\ x_1 & x_2 & x_3 & x_4 & y_5 & y_6 & \dots & y_n \end{array}$$

- Two-point crossover

$$\begin{array}{ccccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & \dots & x_n \\ y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & \dots & y_n \end{array}$$


$$\begin{array}{ccccccccc} x_1 & x_2 & x_3 & y_4 & y_5 & y_6 & \dots & x_n \\ y_1 & y_2 & y_3 & x_4 & x_5 & x_6 & \dots & y_n \end{array}$$

- Uniform crossover

$$\begin{array}{ccccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & \dots & x_n \\ y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & \dots & y_n \end{array}$$


$$\begin{array}{ccccccccc} y_1 & x_2 & y_3 & x_4 & y_5 & x_6 & \dots & y_n \\ x_1 & y_2 & x_3 & y_4 & x_5 & y_6 & \dots & x_n \end{array}$$

Beispiel (applikationsspezifisch)

Finde: $f(x,y) = z$, $x,y,z \in \{T, F\}$

Gegeben: Lernbeispiele der Form $\{T, F\} \times \{T, F\} \times \{T, F\}$ und Regeln:

wenn $x=T$ und $y=F$	dann $z=T$;	wenn $y=T$ dann $z=F$			
10	01	1	11	10	0

Genetische Operationen:

- wähle 2 Positionen $d_1(d_2)$ für die erste Hypothese
- dazu passende Indizes in der zweiten Hypothese
- Crossover

→ Hypothesen unterschiedlicher Länge (Anz. Regeln)

	$(1,8)$	$x \quad y \quad z$	$x \quad y \quad z$
$h_1:$	$1[0 \quad 01 \quad 1$	$11 \quad 1]0 \quad 0$	
	$(1,3)$	$x \quad y \quad z$	$x \quad y \quad z$
$h_2:$	$0[1 \quad 1]1 \quad 0$	$10 \quad 01 \quad 0$	
	$x \quad y \quad z$		
$h_3:$	$11 \quad 10 \quad 0$		
	$x \quad y \quad z$	$x \quad y \quad z$	$x \quad y \quad z$
$h_4:$	$00 \quad 01 \quad 1$	$11 \quad 11 \quad 0$	$10 \quad 01 \quad 0$

Fitness-Funktion: $F(h) = \text{Sum } (\text{correct}(h))$

Hypothese h wird für alle Beispiele ausgewertet

2 Arten der Selektion:

- der Eltern für jeweilige Erzeugung von Nachkommen (*Mating*)
- der Population für die nächste Iteration

Probleme:

- Genetische Drift:
 - Individuen vermehren sich zufällig mehr als andere
- Crowding, Ausreißerproblem
 - „fitte“ Individuen und ähnliche Nachkommen dominieren die Population

→ Entwicklung der Individuen wird verlangsamt
→ Vielfalt der Population wird eingeschränkt

Lösung:

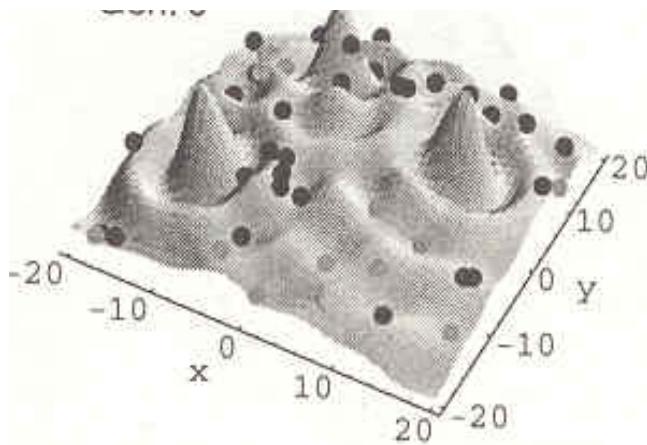
- Verschiedene Populationsmodelle und Selektionsmethoden
- Populationsgröße optimieren

- Inselmodell (lokal)
die Evolution läuft weitgehend getrennt, nur manchmal werden Individuen ausgetauscht
- Nachbarschaftsmodell (nahe Umgebung)
Nachkommen dürfen nur von Individuen erzeugt werden, die in ihrer Nachbarschaft die beste Fitness besitzen
- Eine einfache Menge (global)
die global Besten entwickeln sich rasch weiter, andere Entwicklungslinien werden unterdrückt

Ein Beispiel: Maximumsuche (s. Berthold)

Finde Maximum einer Funktion:

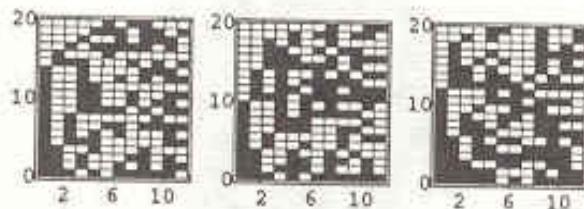
- 3 Populationsgruppen, je 20 Individuen (schwarz, grau, weiß)
- Individuum: binäre Kodierung (12 bit) der x, y – Position
- Operatoren: Mutation / Inversion, 1-point Crossover
- Initiale Verteilung, Gene (siehe unten)



Multimodale Funktion:

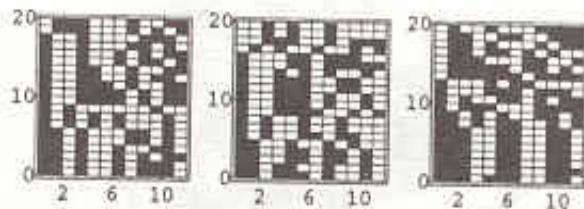
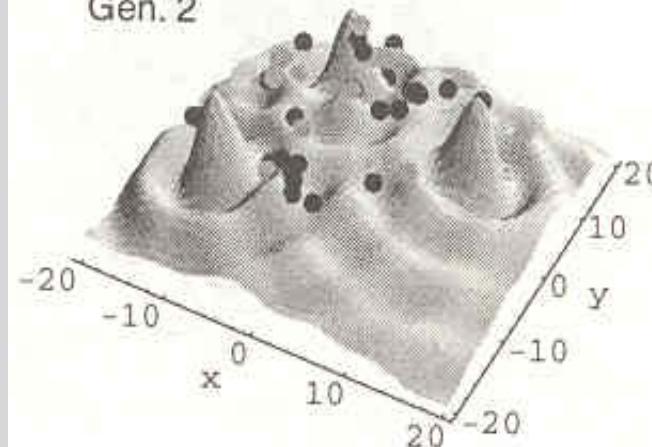
$$f(x,y) = g(x + 11, y + 9) + g(x - 11, y - 3) + g(x + 6, y - 9)$$

$$g(x,y) = \frac{50 \sin(\sqrt{x^2 + y^2})}{\sqrt{x^2 + y^2}} - \sqrt{x^2 + y^2}$$



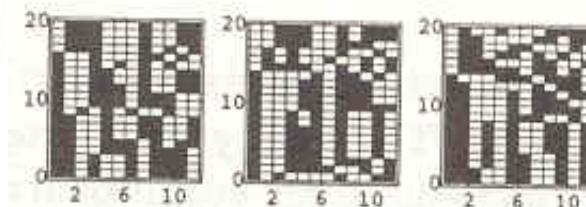
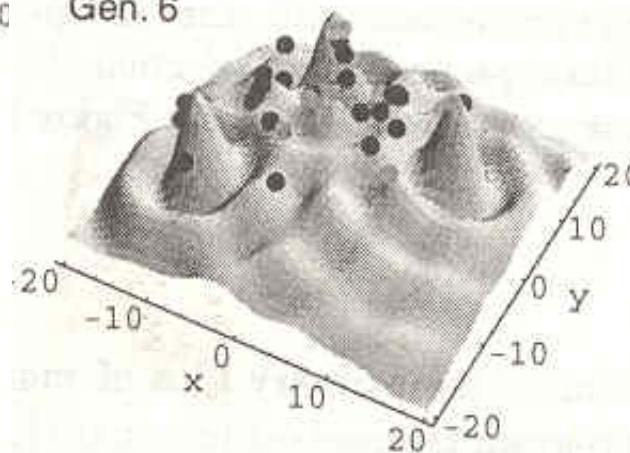
Beispiel: Maximumsuche

Gen. 2

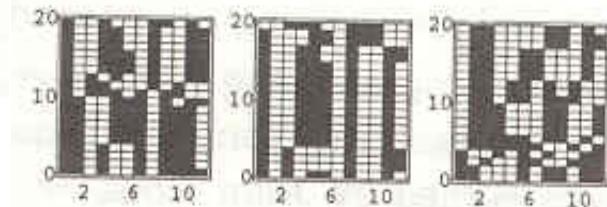
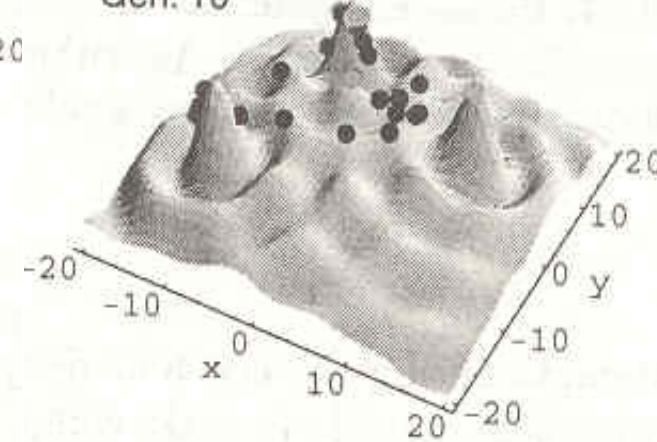


Ziel erreicht nach 10 Iterationen

Gen. 6



Gen. 10



Entwicklung der Gene, Ausrichtung auf das Ziel

Anmerkung: 1 Individuum und Nachbarschaftsmodell → Hillclimbing

Populationsmitglieder (Iteration)

Populationsgröße:

- Soll sie konstant bleiben? (μ)
- Wie viele neu erzeugte Nachkommen? (λ)
- Wie viele Eltern sollen verwendet werden? (ρ)
- Wie werden diese bestimmt?

Mitgliederselektion:

stochastisch ausgewählt \Rightarrow die besten μ Individuen

- (μ, λ) Strategie:
Auswahl bezieht sich nur auf die Nachkommen (bessere Exploration)

- $(\mu + \lambda)$ Strategie:
Auswahl bezieht auch Eltern mit ein (die Besten werden berücksichtigt, Suche nach Eliten \rightarrow Exploitation, günstig bei gut berechenbaren Fitnessfunktionen)

Populationsmitglieder (Iteration)

Ersetzungsregel für Mitglieder:

- Nachkommen ersetzen alle Eltern (Generationen - Modus)
- Nachkommen ersetzen einen Teil der Eltern
- Nachkommen ersetzen Eltern, die ihnen am ähnlichsten sind
- Geographische Ersetzung
- Bestes Individuum überlebt (Elitist – Modus)

Daumenregel [Braun]:

Das beste Viertel der Population sollte drei Viertel der Nachkommen erzeugen

Fitness Based Selection

$$P(x) \approx \frac{f(x)}{\sum_{x' \in Pop.} f(x')} \quad \text{genau}$$

$$P(x) = \frac{\lambda}{\mu} \cdot \frac{f(x)}{\sum_{x' \in Pop.} f(x')}$$

$P(x)$: Wahrscheinlichkeit der Auswahl von Individuum x

λ : Anzahl von Nachkommen

μ : Populationsgröße

f : Fitness - Funktion

-abhängig vom Wert der Fitnessfunktion → Problem wenn z.B. im Laufe der Evolution nur noch geringe Änderungen in $f(x)$ und damit in $P(x)$

Selektionsmethoden

Ranking Based Selection

$$P(x) \approx \frac{g(r(x))}{\sum_{x' \in Pop.} g(r(x'))} \quad mit$$

$P(x)$: Wahrscheinlichkeit der Auswahl von Individuum x

$r(x)$: ranking von x in der aktuellen Population gemäß Fitness - Funktion

g : mit der Güte des Ranges monoton steigende Funktion
größer 0

- Exponentiell: $g(x) = a^{-x}$

- Hyperbolisch: $g(x) = x^{-a}$

- Die besten k :
$$g(x) := \begin{cases} 1/k & x \leq k \\ 0 & sonst \end{cases}$$

- weniger anhängig von dem Betrag der Fitness
- bessere Anpassung von Exploration / Exploitation durch g

Tournament Selection (Turnier)

- wähle für jedes zu erzeugende Individuum $n (=2)$ Individuen
- belohne (Bewertung erhöhen) davon, das gemäß der Fitness beste Individuum
- wähle Individuen mit höchster Bewertung
- wenig anhängig von dem Betrag der Fitness

Wahl der Selektionsmethode

- oft anwendungsspezifisch

Lamarksche Evolution

- Individuen ändern sich (lernen) nach der Erzeugung
- Genotyp (alle Gene) wird verändert und anschließend vererbt

Baldwinsche Evolution

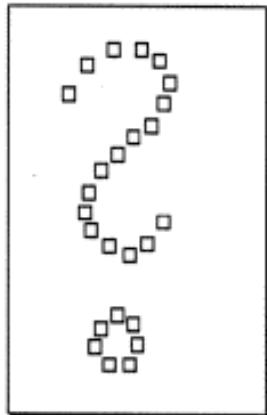
- Individuen ändern sich (lernen) nach der Erzeugung
- Genotyp bleibt unverändert

Hybride Verfahren

- es gibt veränderbare und fixe Phänotypen

Anwendung: Suche nach optimalen neuronalen Netzen

Beispiel: Travelling Salesman-Problem

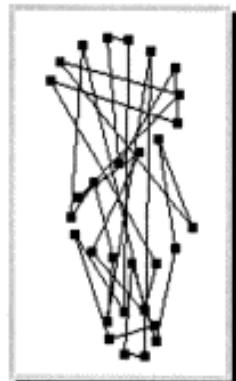


Finde einen Pfad:

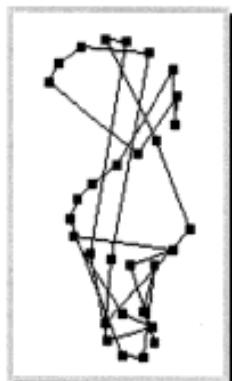
- jeder Ort wird genau einmal besucht
- der zurückgelegte Weg ist minimal

Lösung:

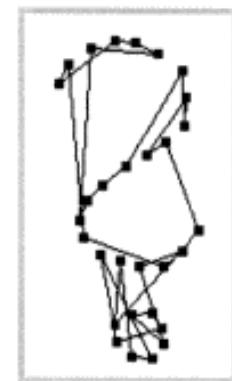
- Evolutionsstrategie → 22 Iterationen



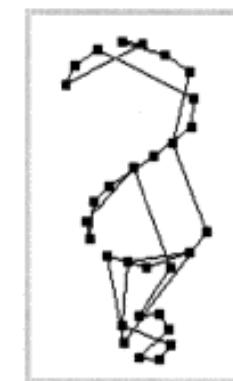
G=0
Q=1958.9



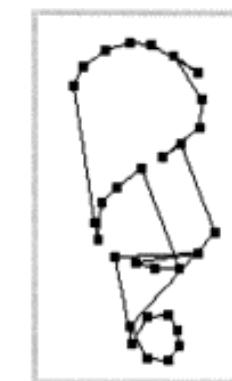
G=1
Q=1191.5



G=2
Q=1003.0



G=3
Q=881.8



G=4
Q=713.9

Travelling Salesman

Fitnessfunktion:

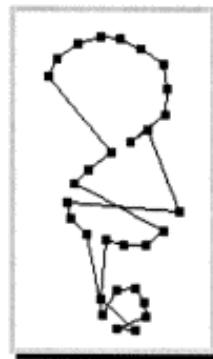
Länge des Weges von Start zu Ziel aber ranking-based,
da bereits ab wenig Iterationen Diff. der Fitness unter 5%

Kodierung und Mutationen:

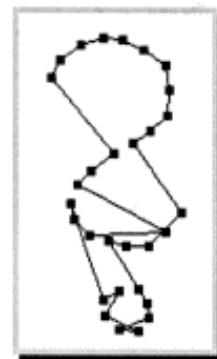
1 2 3 4 5 6 7 8 9 10 11 12	Parents-Tour
1 5 4 3 2 6 7 8 9 10 11 12	Inversion of the tour segment 2-3-4-5
1 3 4 5 2 6 7 8 9 10 11 12	Insertion of town no.2 between no.5 and no.6
1 6 7 2 3 4 5 8 9 10 11 12	Displacement of the tour segment 2-3-4-5
1 5 3 4 2 6 7 8 9 10 11 12	Reciprocal Exchange of towns no.2 and no. 5

The mutation operators for the TSP

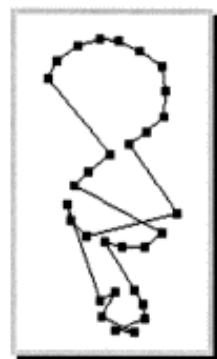
Travelling Saleman



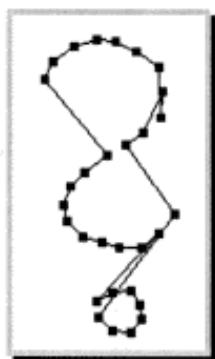
G=5
Q=644.0



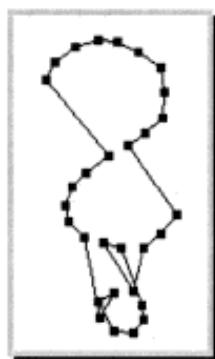
G=7
Q=627.6



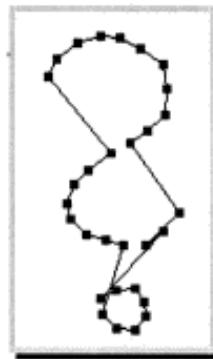
G=8
Q=624.1



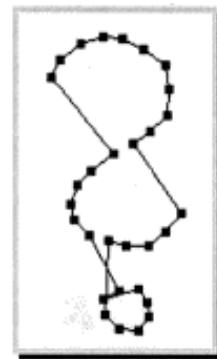
G=9
Q=575.3



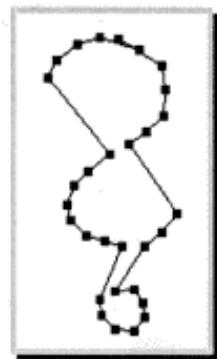
G=10
Q=554.6



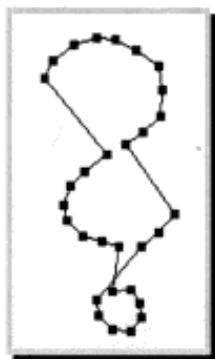
G=11
Q=529.5.0



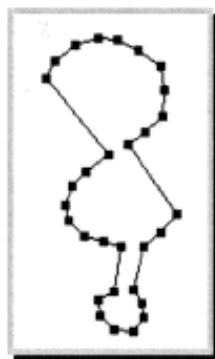
G=12
Q=512.7



G=15
Q=511.8



G=16
Q=487.5



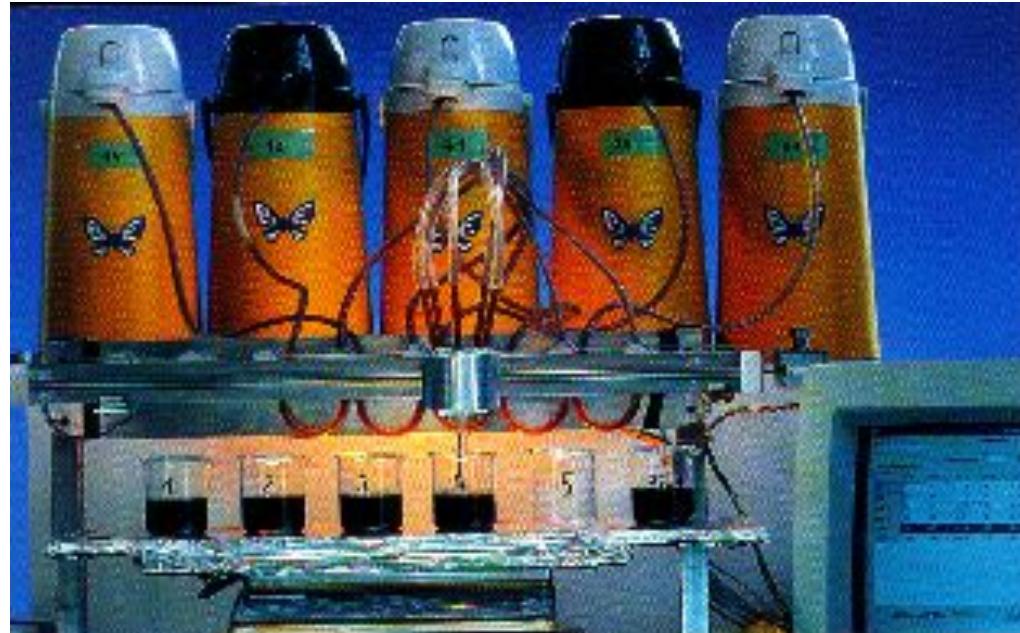
G=22
Q=472.0

Anwendungsbeispiel

Anwendung: Flugplanoptimierung



Mischung von Kaffeesorten (Herdy98)



Problem:

Markenkaffee sollte immer gleich schmecken, obwohl es immer wieder zu unterschiedlichen Mischverhältnissen kommt.

Übliches Vorgehen:

Experten mischen solange, bis die Mischung den gewünschten Geschmack hat.

Mischung von Kaffeesorten (Herdy98)

Fitnessfunktion:

Mensch - Geschmackstester

Test der Evolutionsstrategie:

- anfangs ganz unterschiedlich schmeckende Mischungen.
- nach 11 Generationen können Experten keinen Geschmacksunterschied mehr entdecken

Vorteil:

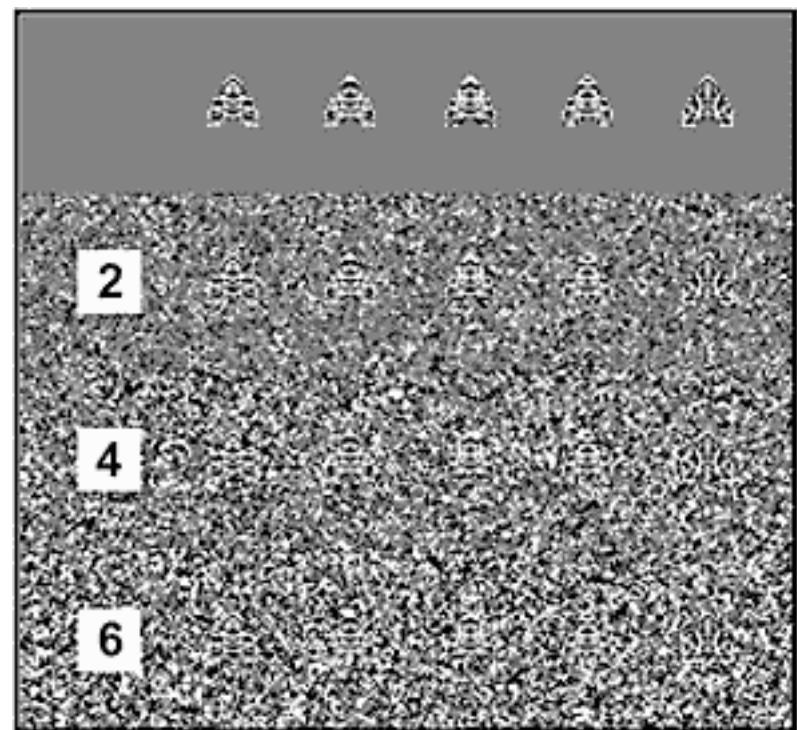
Auch Kosten und andere Optimierungskriterien können einbezogen werden

Anwendungsgebiete:

- Mischungsgetränke wie Kakao, Tee, Whisky
- Kompositionen wie Fliesenfarben, Farbtöne etc
- Kriminalistik – Personen identifizieren

Cybermotten [Univ. Nebraska]

- Wer sich tarnt, wird seltener gefressen. Was aber ist eine gute Tarnung?



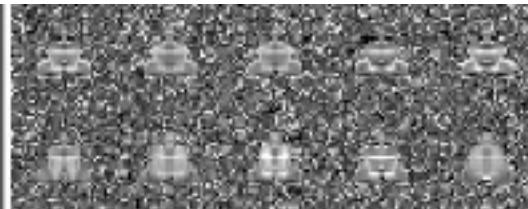
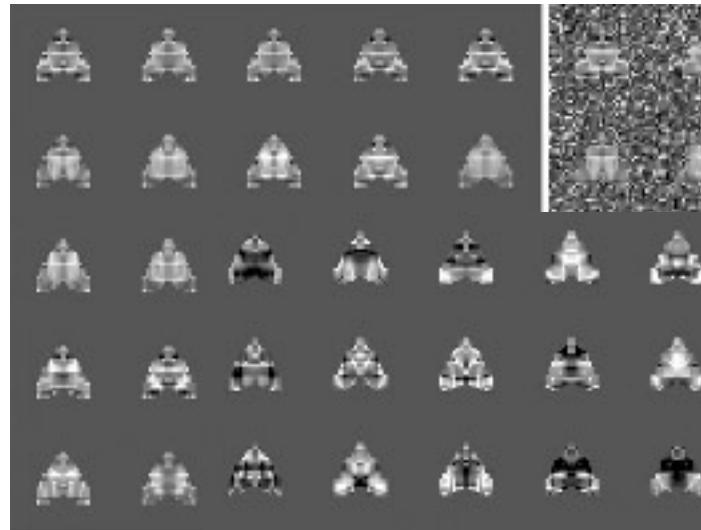
Cybermotten

- Konstante Population 200
- Fitness – Zeit in der die Motten vom Vogel entdeckt werden
- Rekombination und Mutation
(auf dem Graycode der Grauwertpixel = 117 Bytes)
- Eltern erzeugen mehrere Nachkommen
- Eltern „überleben“ nur eine Generation

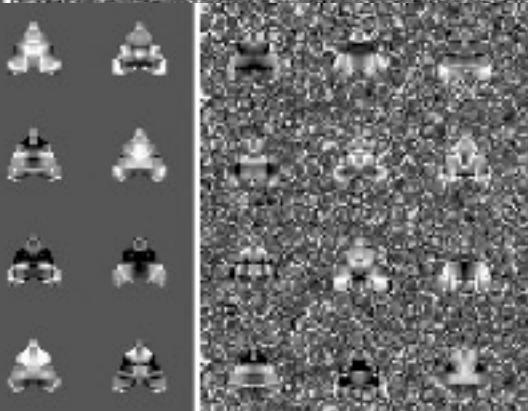


Cybermotten – Entwicklung

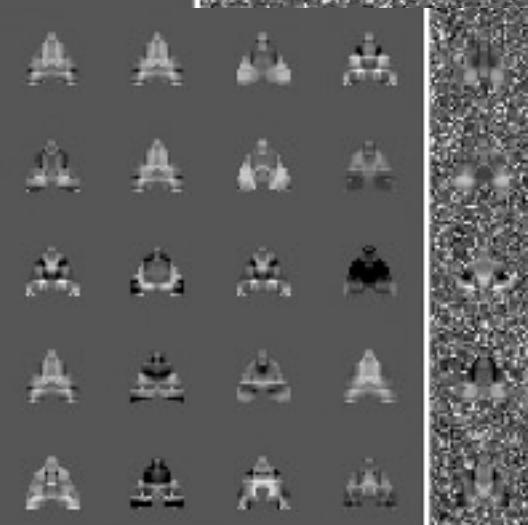
■ Start



■ Nach 100 Gen.



■ Fiteste Individuen
– Anpassung

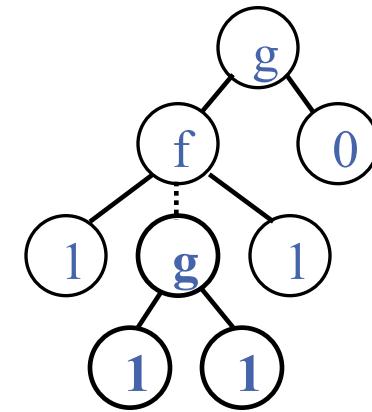


Genetische Programmierung

Ziel: Erzeugung optimierter Programme

- Individuen sind Programme
- Repräsentation z.B. als Baum
- Selektion, Mutation und Rekombination auf Baumstrukturen
- Fitness: Programmtest auf einer Menge von Testdaten

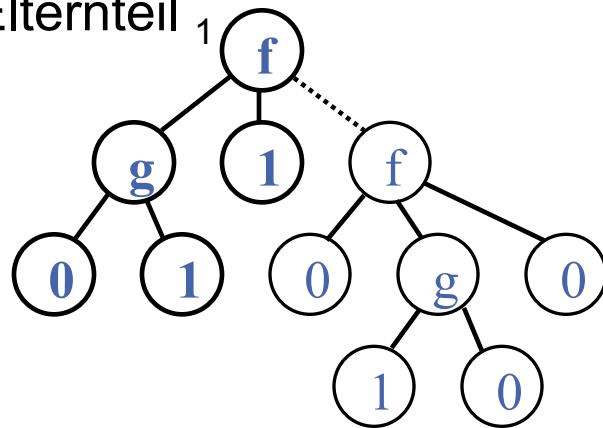
$g(f(1..g(1,1)..1,0)$



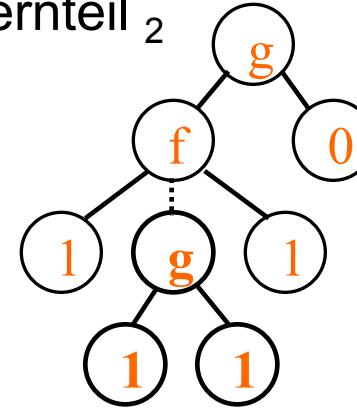
Genetische Programmierung: Rekombination

Rekombination als Austausch von Teilbäumen

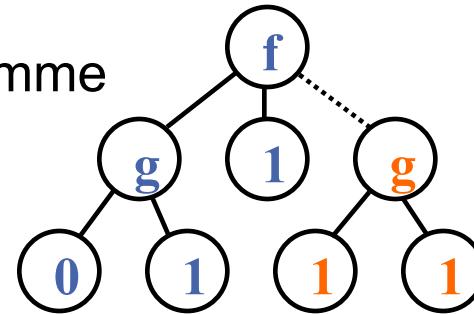
Elternteil 1



Elternteil 2



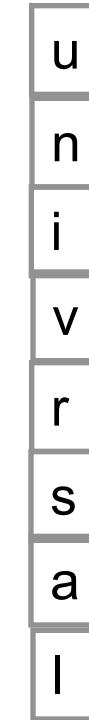
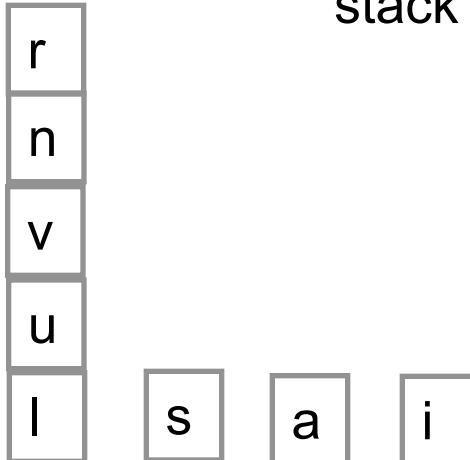
Nachkomme



Genetische Programmierung: Beispiel

Ziel:

durch pick and place
das Wort „universal“ am
stack erzeugen



Genetische Programmierung: Beispiel

Individuen bestehen aus folgenden Operationen:

EQ(x,y):	return $x == y$
DU(x,y):	do x until y
MT(x):	if x on stack: move x to table, return true else return false
CS:	return current stack element
NOT(x)	inverse x
MS(x):	if x on table move x to stack, return true else return false
NN:	return next necessary on stack

Fitness:

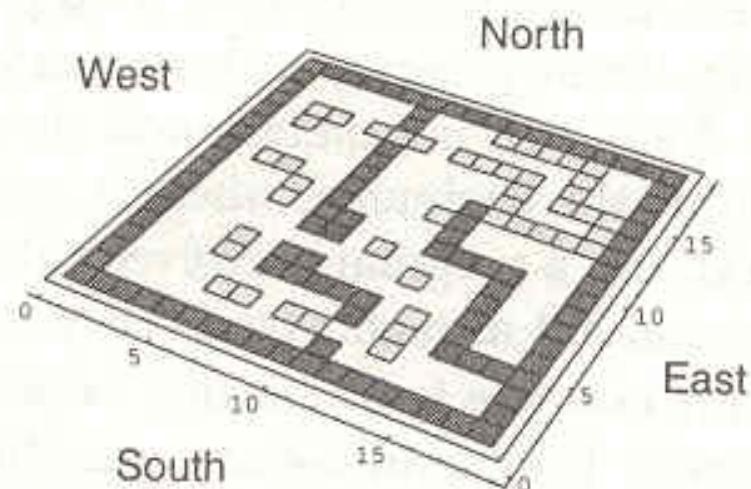
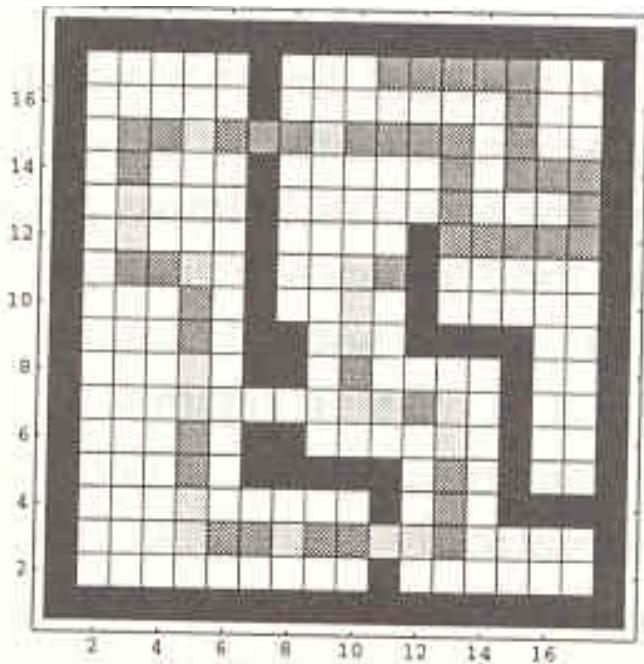
Test auf 166 Startkonfigurationen (Block-Verteilungen)

Ergebnis:

EQ ((DU (MT CS) (NOT CS)) , (DU(MS NN), (NOT NN)))

Beispiel: Ameise [Berthold]

Ziel: Programm (Steuerung) einer Ameise, s.d. mit Start von (10,17) → gesamtes Futter, wegoptimal gesammelt wird.



Zellen: Wand (schwarz), Futter (dunkelgrau), Pheromone (hellgrau)

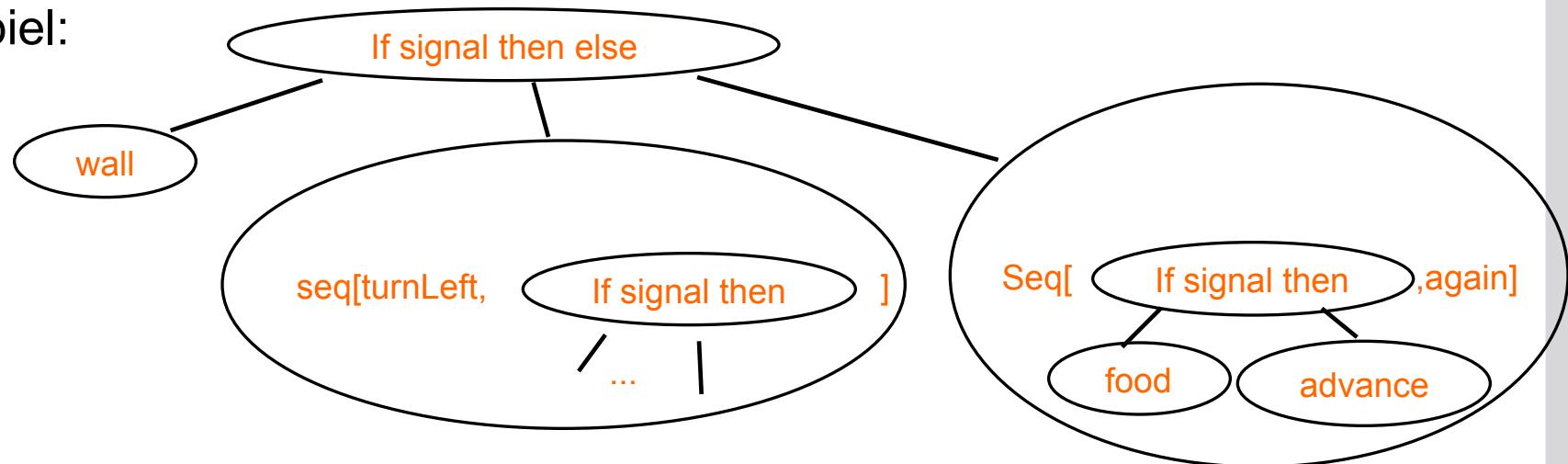
Beispiel: Ameise

Funktionen: if signal then; if signal then - else
 seq (Sequenz von Operatoren), again(letzte if-Anweisung)

Signal: food, wall, pheromone, dust

Steuerung: advance, turnLeft, turnRight, nop

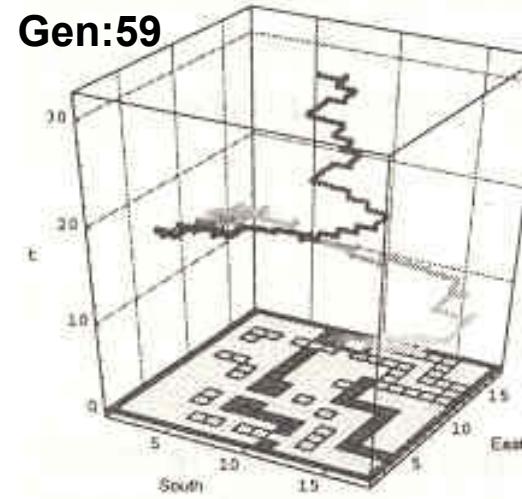
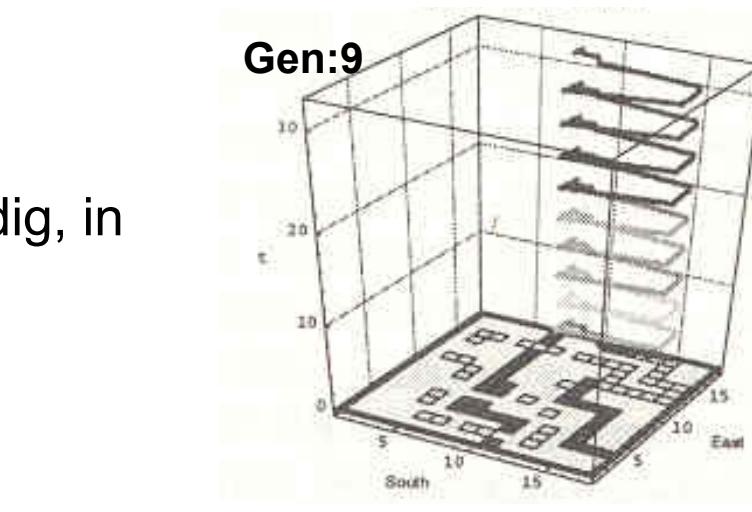
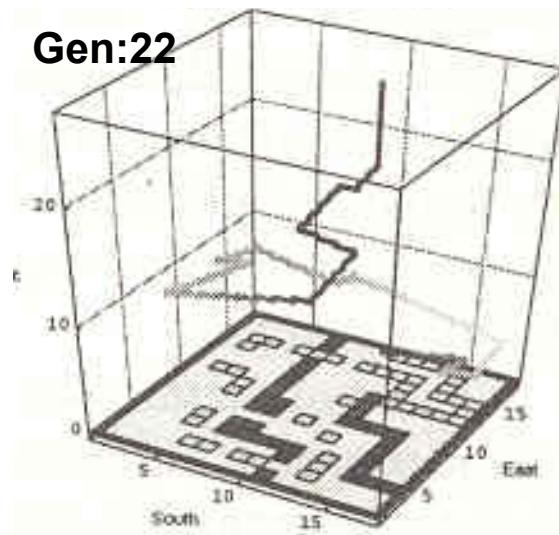
Beispiel:



Programm: z.B. zyklischer Ablauf

Beispiel: Ameise

Nach 59 Iterationen optimales Programm, Futter wird vollständig, in kürzester Zeit gesammelt



↑
Programmablauf

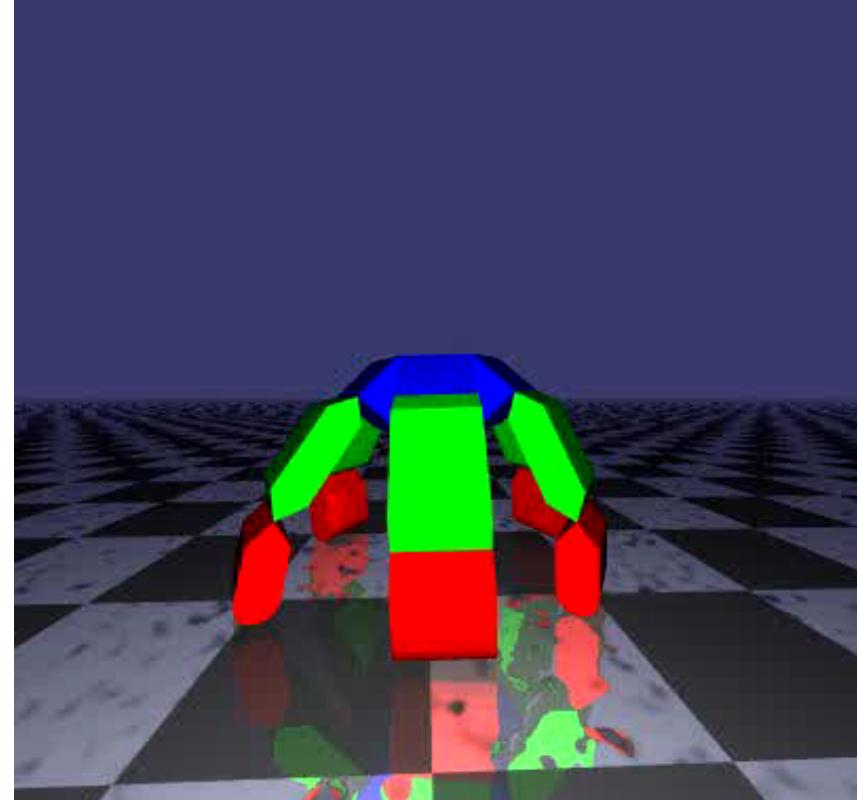
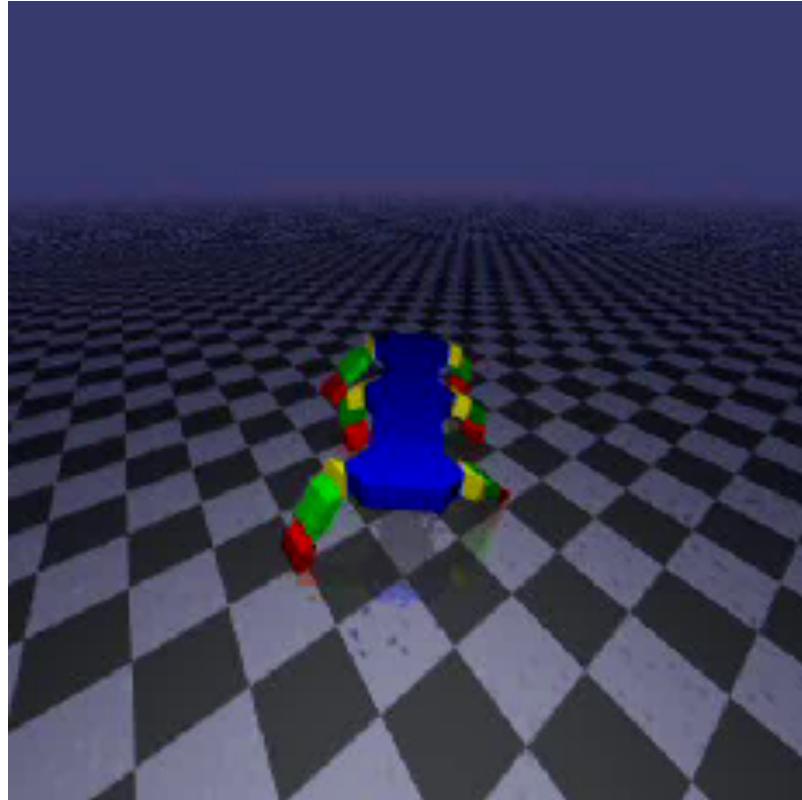
Steuerung in der Robotik – Optimale Steuerung [Ziegler]

- Optimale Steuerung von Laufmaschinen mit unterschiedlicher Morphologie
- Beispiel – Erzeugung einer Beinsteuierung
- Steuerungsparameter: Bewegung pro Gelenk
 - Gelenknummer $integer[0;3]$
 - Startzeitpunkt $real[0;t]$
 - Dauer $real[0;t]$
 - Kräfte $real[-max;max]$

Genetisches Programmsystem

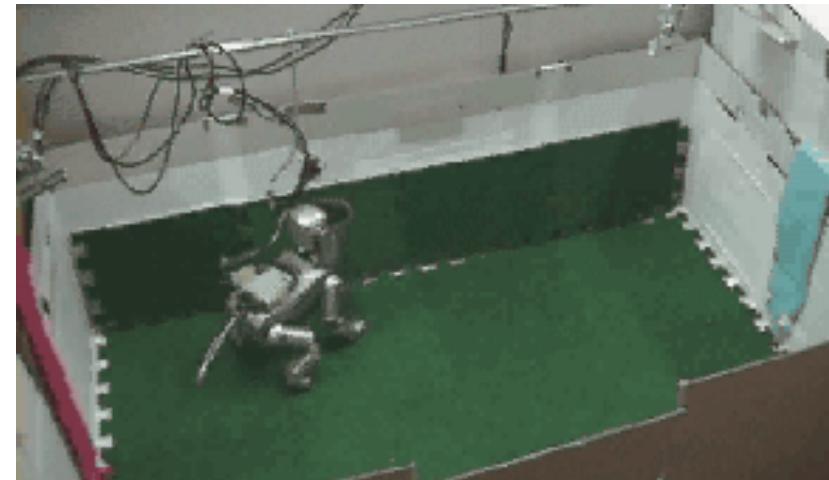
- Gegeben: Kinematisches System
- Individuum = Sequenz von Instruktionen variabler Länge
- Mutation
 - Micro-Mutation (Veränderung einzelner Werte einer Instruktion)
 - Macro-Mutation (Austausch ganzer Instruktionsblöcke)
- Anpassung
 - Crossover
 - Micro-Crossover (Änderung eines Parameters einer Instruktion)
 - Macro-Crossover (Austausch einer Instruktion)
 - Homologes-Crossover (Austausch von Instruktionen eines speziellen Gelenks)
- Fitness = Abweichung von einer vorgegebenen Trajektorie

Beispiel der Beinsteuerung in einer Simulation



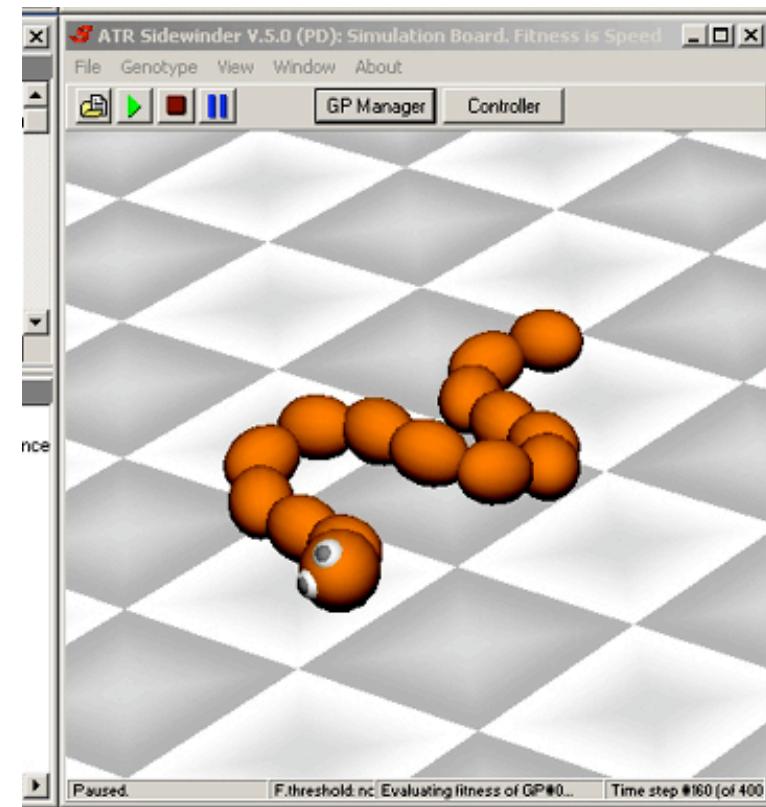
Autonomous Evolution of Dynamic Gaits[Pollack]

- Ähnlicher Ansatz für AIBO
- Fitness = Mittelwert aus drei Läufen; Funktion aus Richtung und Geschwindigkeit
- Lernen
- Ergebnis



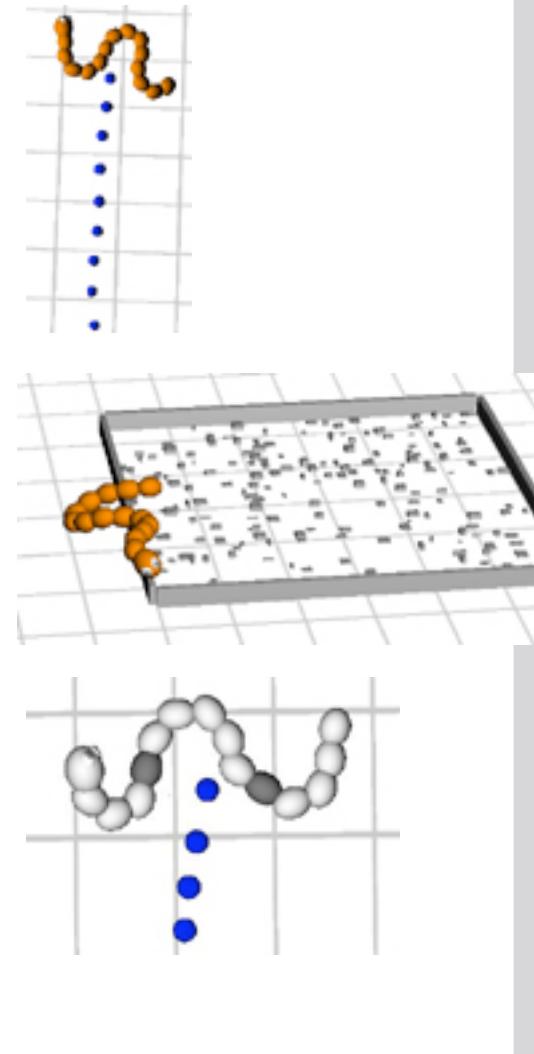
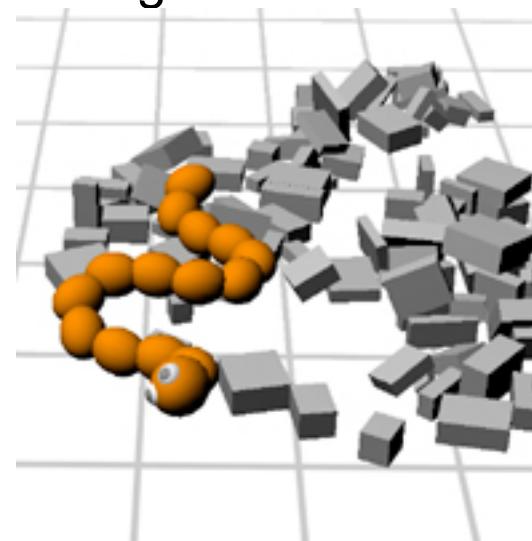
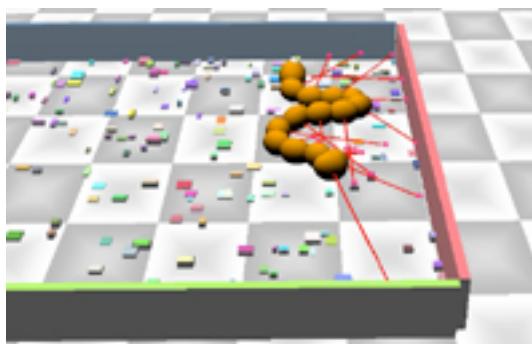
Snakebot [Ivan Tanev, Doshisha University Japan]

- Biologische Inspiration: Seitenwinder Klapperschlange (*Crotalus Cerastes*) 5 km/h und die schnellste Schlange , die Schwarze Mamba (*Dendroaspis Polylepis*) - 16 km/h



Snakebot

- Individuum = Steuerung
(Programm = Zeitreihe für jedes Gelenk)
- Fitness = Geschwindigkeit in
 - Fortbewegung
 - „Klettern“, Befreien aus Hindernissen etc...
- Adaptation: ein/zwei Gelenke unbeweglich
- Erweiterung Hinderniserkennung



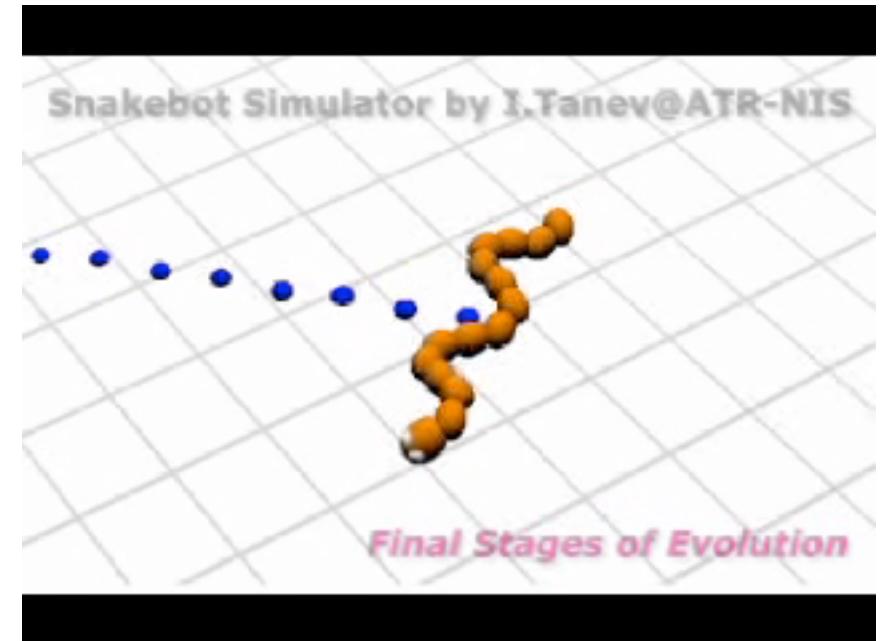
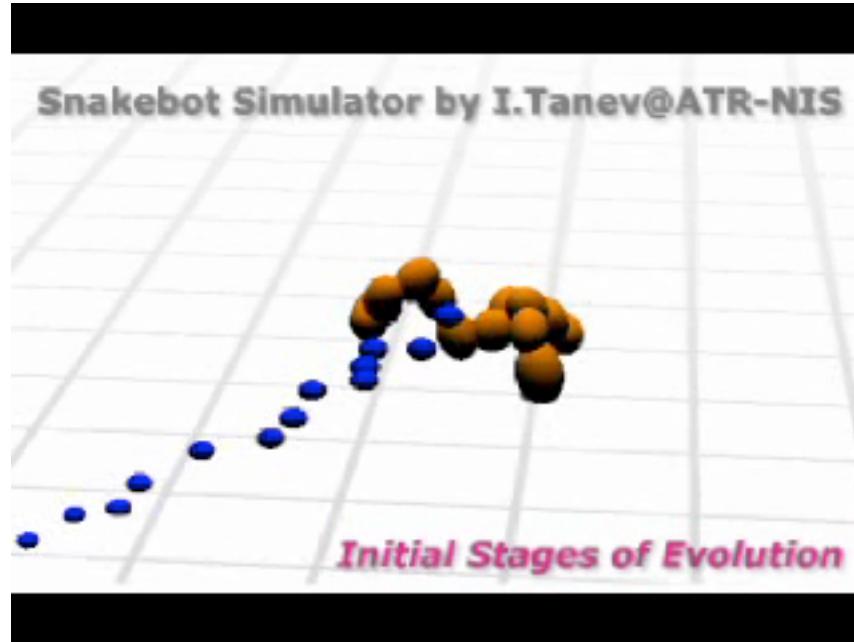
Genetisches Programm [I. Tanev 2005]

TABLE I
MAIN PROPERTIES OF GP

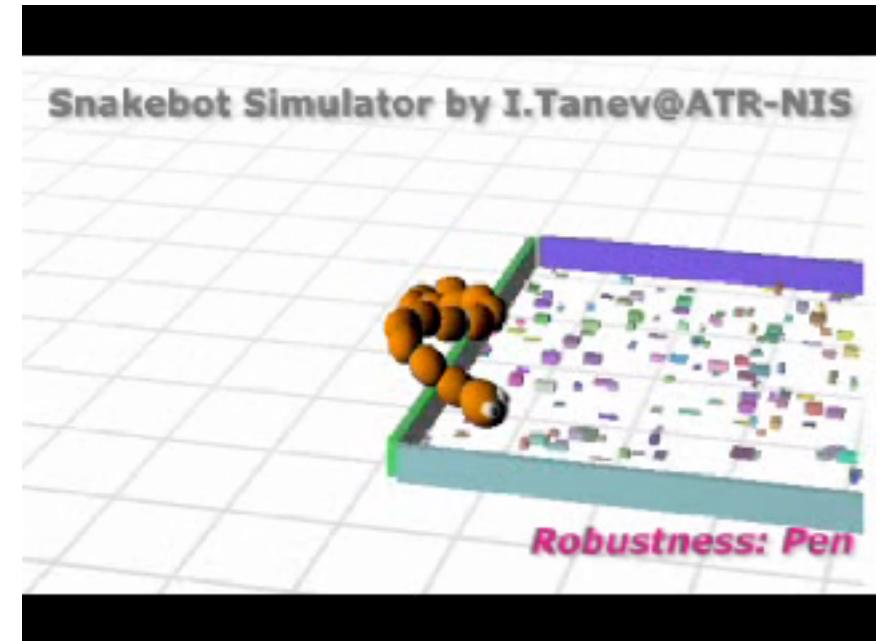
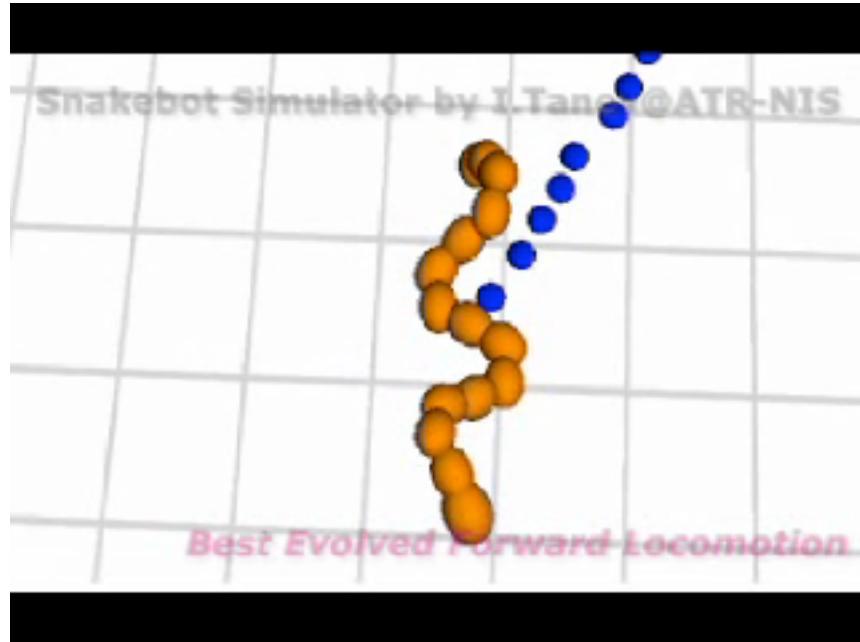
Category	Value
Function set	{sin, cos, +, -, *, /}
Terminal set	{time, segment_ID, Pi, random constant, ADF}
Population size	200 individuals
Selection	Binary tournament, ratio 0.1
Elitism	Best 4 individuals
Mutation	Random subtree mutation, ratio 0.01
Fitness	Velocity of simulated Snakebot during the trial
Trial interval	180 time steps, each time step account for 50ms of “real” time
Termination criterion	(Fitness >100) or (Generations>40) or (no improvement of fitness for 16 generations)

Snakebot

■ Evolutionsschritte

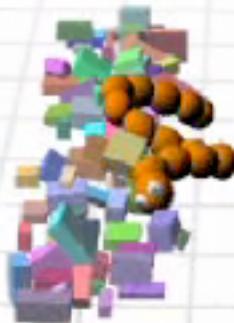


Snakebot



Snakebot

Snakebot Simulator by I.Tanev@ATR-NIS



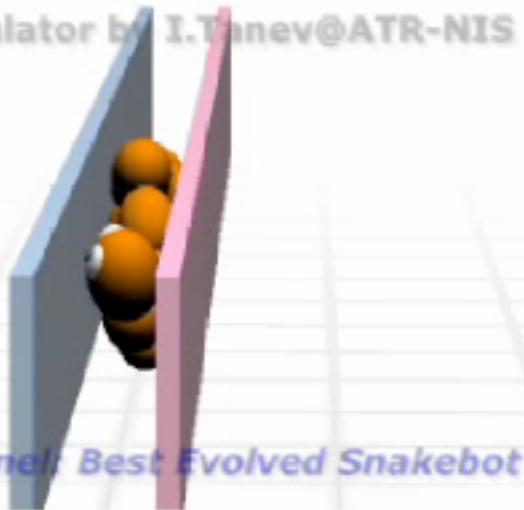
Robustness: Pile of Boxes

Snakebot Simulator by I.Tanev@ATR-NIS



Robustness: Stack of Boxes

Snakebot Simulator by I.Tanev@ATR-NIS

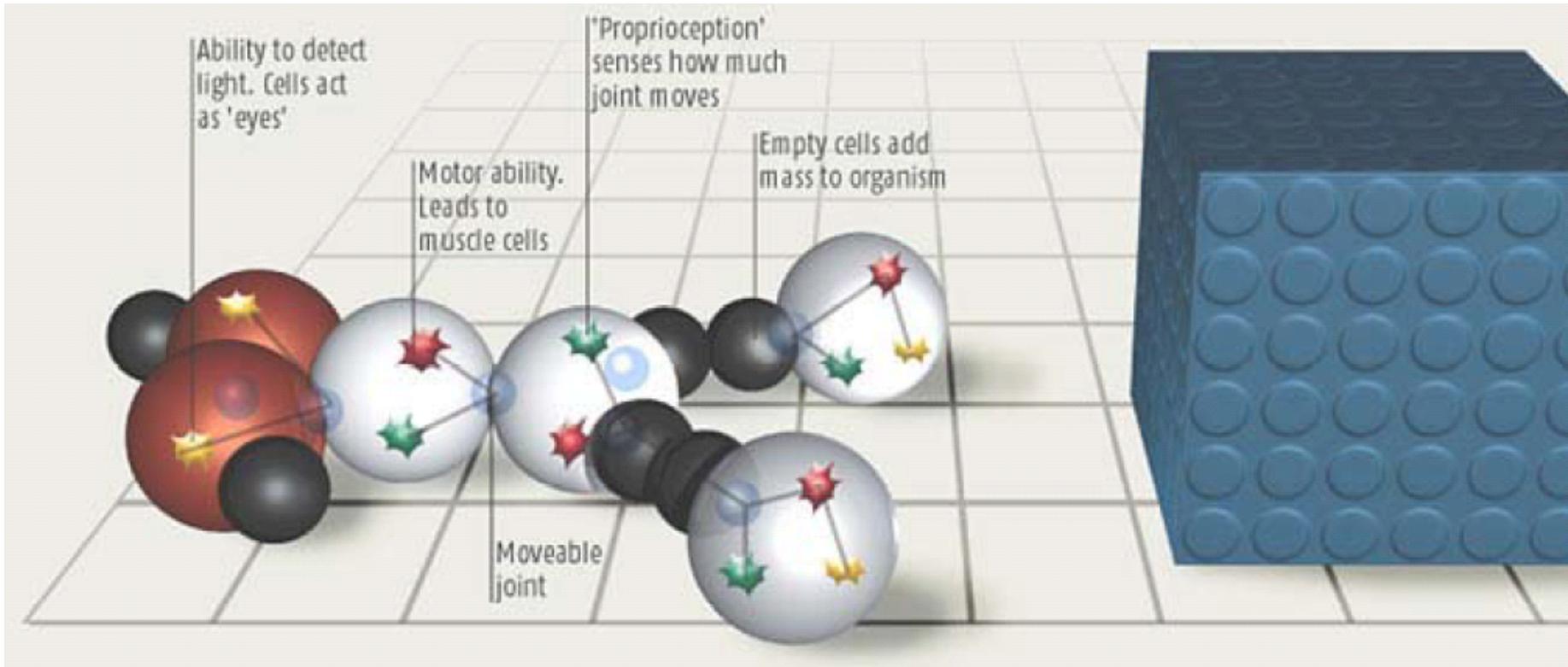


Tunnel: Best Evolved Snakebot

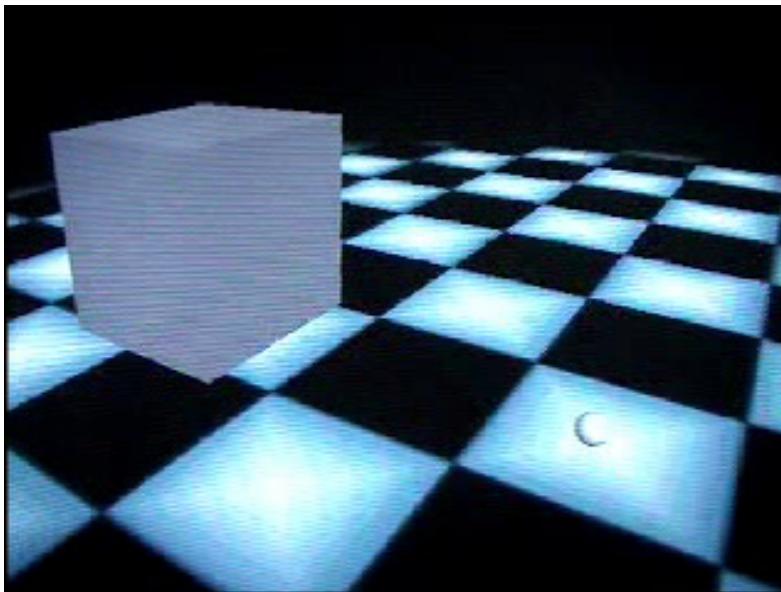
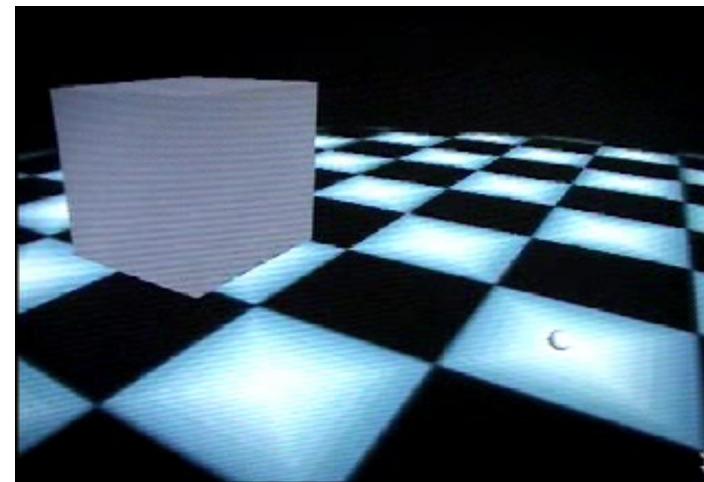
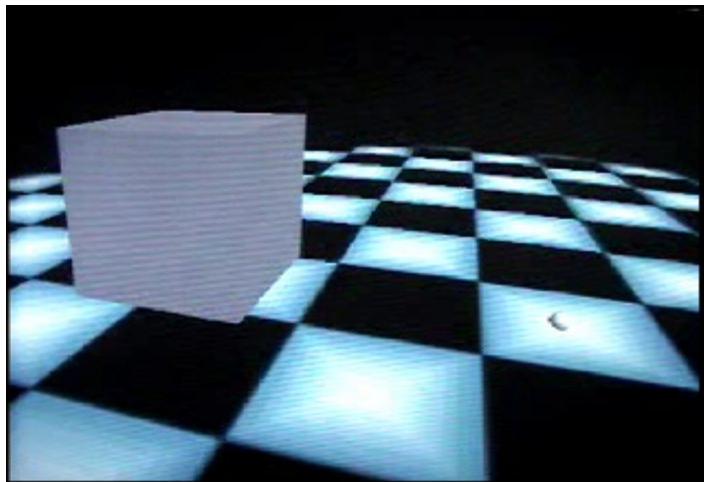
Künstliche Ontogenese [Bongard]

- Entwicklung von Agenten durch „künstliche Zellen“
- Repräsentation einer Zelle durch Kugel
- Operatoren
 - Wachsen der Kugeln im Durchmesser,
 - bei Erreichen der maximal Größe - Teilen in zwei Kugeln, die z.B. durch Rotationsgelenk oder ein starres Element verbunden sind
- Kugeln besitzen ggf. ein neuronales Netz mit Motoneuron, Interneuronen und Perzeptionsneuron
- Kodierung als Gleitkomma Genom

Künstliche Ontogenese – Prinzip



Künstliche Ontogenese – Experimente



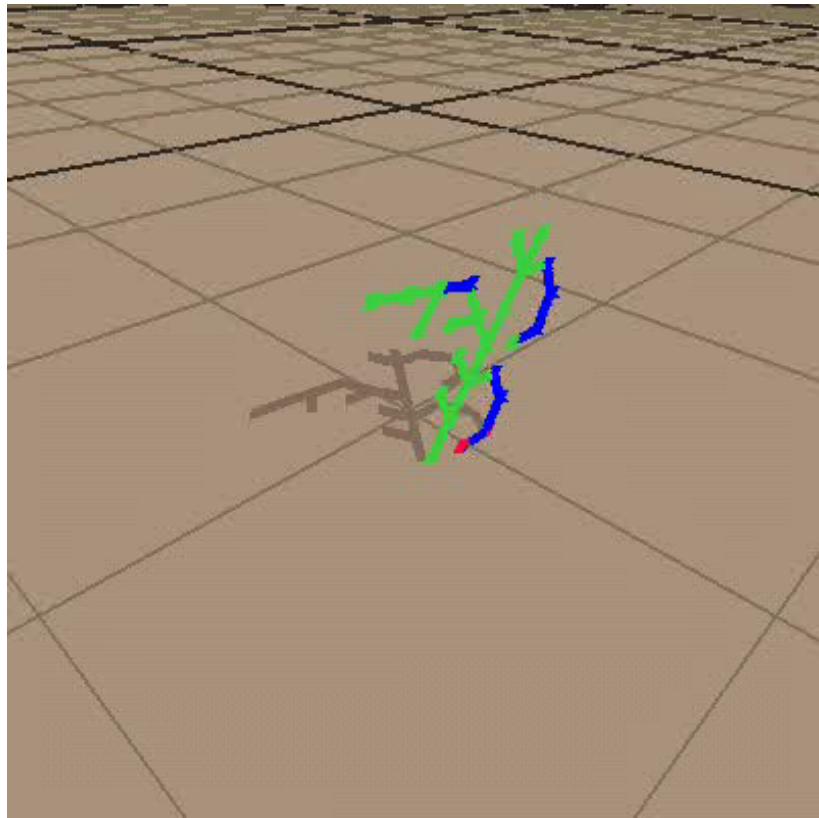
Fitness: Bewegung des Blocks (in 1000 Zeiteinheiten, mehrfache Versuche)

Selektion: Elitist 1 (beste Individuum in nächste Population)

Operatoren: Wachstum / Änderung anschließend Mutation (Crossover)

Genobots – Automatisches Modulares Design [Hornby]

- Blau – Aktuator (eigenständig oszillierend), Grün - steif

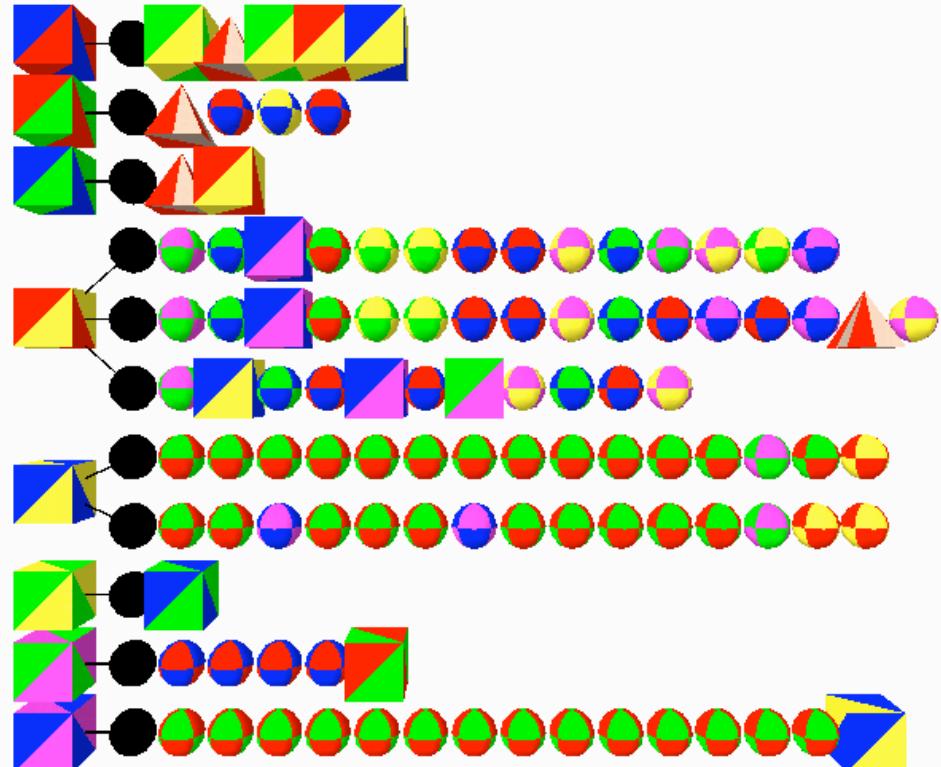


Encoding for Automated Design

An evolved creature.

Genobots – Automatisches Modulares Design [Hornby]

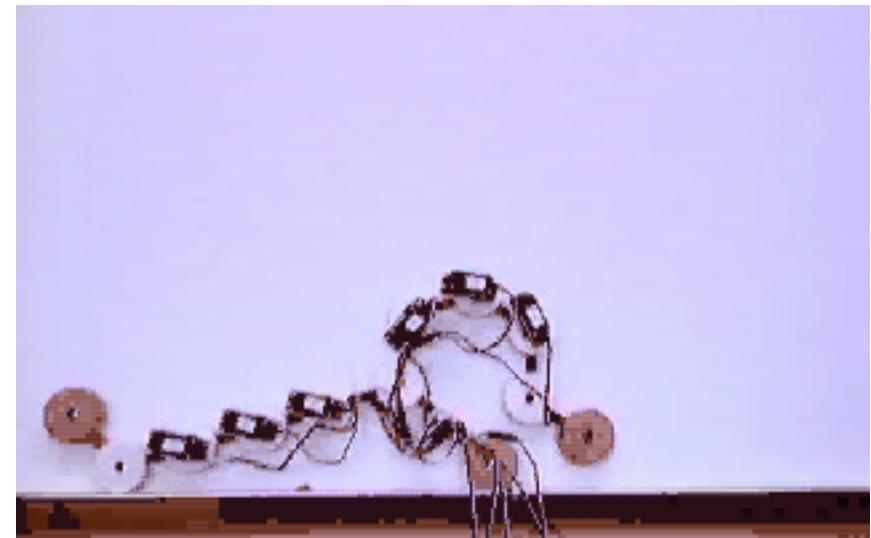
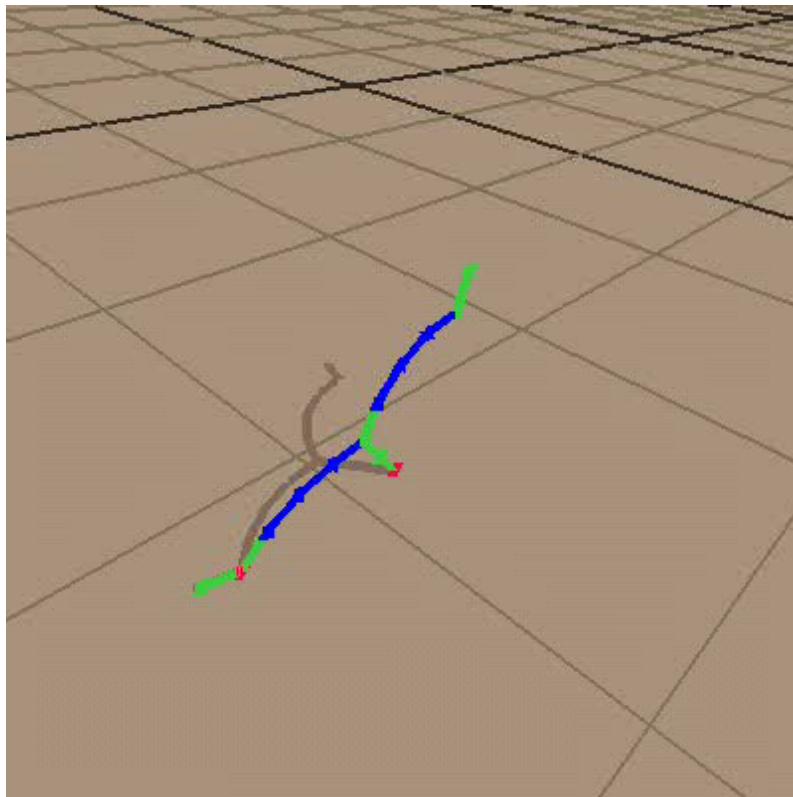
- Individuen = Regeln und Algorithmen für die Erzeugung (Generative Repräsentation) von künstlichen Systemen
- Anw.: Mobile Systeme
- Baupläne als Gramatik:
 - Gelenke
 - Verbindungen
 - Parameter
 - ...
- Steuerung: Jedes Gelenk oszilliert eigenständig



Genetischer Algorithmus

- Population 100 Individuen
- Evolution: 500 Iterationen
- Genetische Operatoren
 - Mutation: Parameter und Kommandoblöcke
 - Rekombination: Ersetzten von Teilsequenzen,....
- Fitness: Distanz die jedes Individuum in der Simulation in 10 Zyklen zurücklegt (normalisiert)
- Fitness-exponentielles scoring (ranking) Faktor 0,03 für die Populationsselektion
- Mating: Gleichverteilt
- Elitism 2 – Methode (die zwei Besten bleiben erhalten)

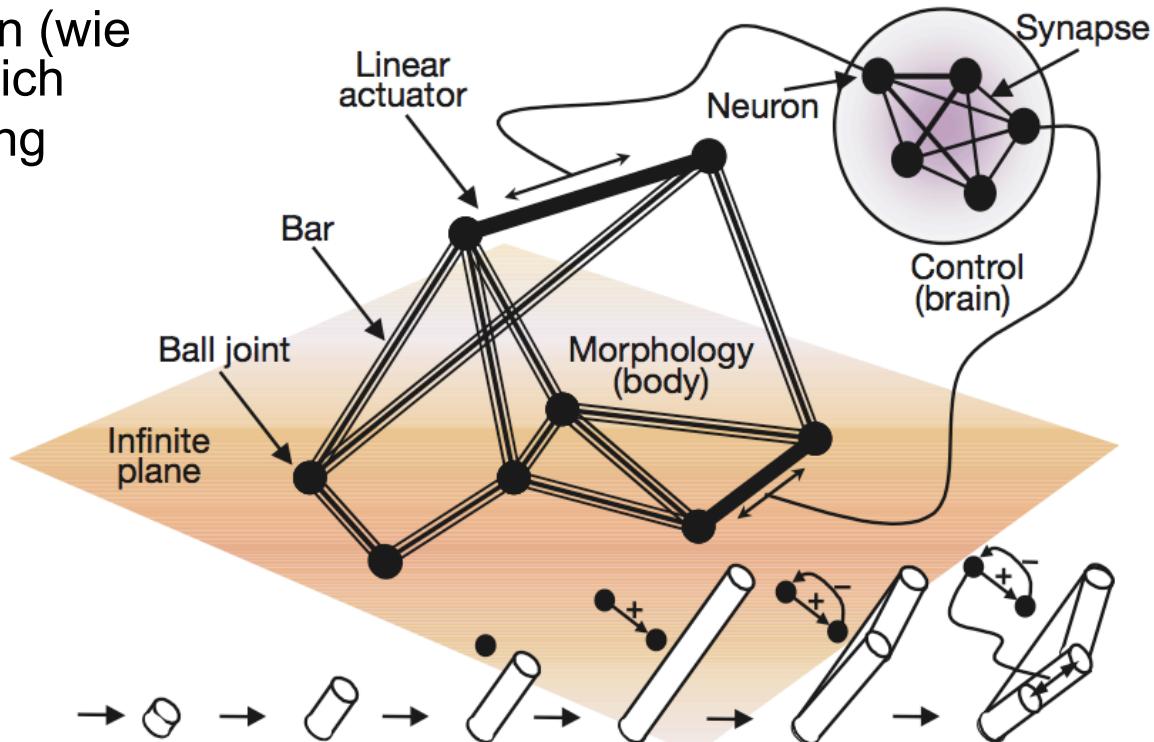
Genobots – Automatisches Modulares Design



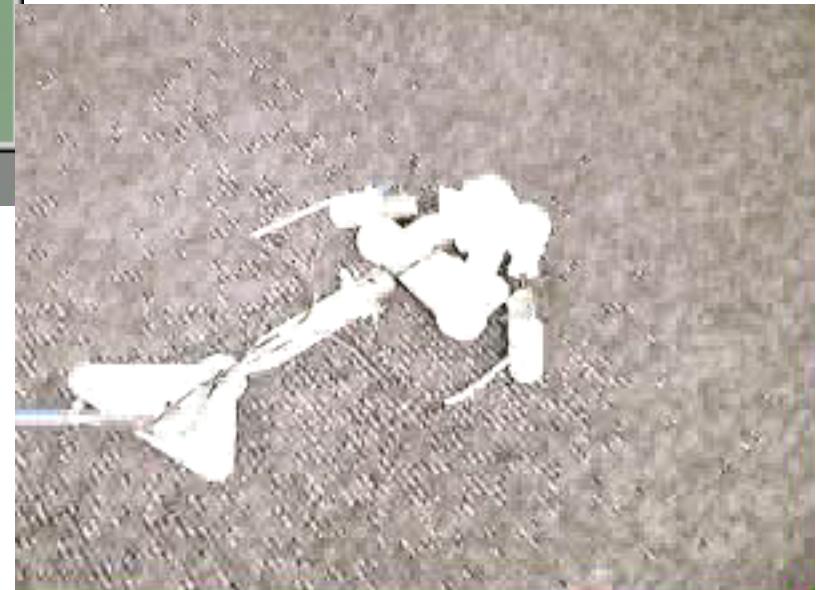
Reale Umsetzung

Golem - "life as it could be" [Pollack]

- Individuum = Repräsentation der Morphologie
- Roboter: Stangen (Länge, D, Steifigkeit), Aktuatoren, Gelenktypen
Neuronen (threshold, synaptische Verbindung)
- Beschränkungen
 - Komplexe Strukturen (wie Muskeln) nicht möglich
 - Design Beschränkung
 - Evaluation (Fitness)
 - Übertragbarkeit in die Realität
- Evolution ähnlich Genobots
 - Gelenke
 - Neuronen,...



Golem - "life as it could be"



Reale Umsetzung

Weitere Anwendungen

- Optimierung der Topologie Neuronaler Netze
- Optimierung des Muskel-Skelett Systems hinsichtlich energieeffizienter Steuerung
- Optimierung der Gangarten von Laufmaschinen hinsichtlich Stabilität
- Reduzierung des Luftwiderstands durch Tragflächenspitzen beim Flugzeug
- Platinenlayout
- Multicore Programmierung

- ☺ Add – in für Microsoft Excel – Optimierung von Funktionen (<http://www.neuralmarkettrends.com/2007/11/11/genetic-algorithm-excel-addin/>)

Diskussion

- Gute Parallelisierbarkeit
 - Parallele Suche
 - Population von Individuen \Leftarrow Mehrere erfolgversprechende Lösungsrichtungen sind möglich

- Ansätze meist sehr rechenintensiv
- Kombination mit Multi-Core, GPU und FPG Implementierungen sind Gegenstand aktueller Forschung
- Optimierungsprobleme in der Simulation lösbar
- Integration von Vorwissen möglich durch Initialisierung der Population
- Um strukturelles und anwendungsspezifisches Wissen einsetzen zu können, müssen allgemeine Algorithmen spezialisiert werden.

Literatur

- [1] *Tom Mitchell: Machine Learning.* McGraw-Hill, New York, 1997.
- [2] *Michael Berthold, D.J. Hand: Intelligent Data Analysis.* 2nd Edition, Springer, 2003.
- [3] *H. Braun: Neuronale Netze - Optimierung durch Lernen und Evolution.* Springer, 1997.