# 簡易コーディングを用いた進化型ニューラルコントローラの性能評価

摂南大 ○ 片田 喜章，高田 直樹

# Investigation of Evolutionary Neural Controllers with Simple Coding for a Mobile Robot

Yoshiaki KATADA* and Naoki Takada*
*Faculty of Engineering, Setsunan University

**Abstract:** One of the advantages of evolutionary robotics over other approaches in embodied cognitive science would be its parallel population search. Due to the population search, it takes a long time to evaluate all robot in a real environment. Thus, such techniques as to shorten the time is required for real robots to evolve in a real environment. This paper proposes to use evolutionary neural controllers with simple coding to make genetic search space small and investigates performances of them on an evolutionary robotics task using a simulated robot. The results suggest the benefits of the proposed method.

## 1. はじめに

ロボットが実環境で振舞うためには，システム自身が高い環境適応能力を有し，環境との相互作用を通して自らの振舞いを獲得しなければならない [1]．このアプローチの一つに多点探索による高い解探索能力が期待できる人工進化を用いて自律ロボットの制御器を設計する進化ロボティクス (Evolutionary Robotics:ER)[2] がある．進化ロボティクスでは，制御器の評価は実環境で実ロボットを用いて行うか [3]，または，これらのシミュレーションを用いて行われる [4]．これらの方法はそれぞれ問題点が指摘されている．実環境で実ロボットを評価するには当然，実時間がかかる．人工進化で要する進化時間は数百世代から数千世代が一般的であり，膨大な時間がかかる．しかしながら，ロボットは振舞うべき実環境で評価されているため，その環境で適切に振舞うことができる．一方，シミュレーションを用いる方法では，ロボットの評価にかかる時間は実環境に比べて非常に小さいことは言うまでもない．しかしながら，シミュレーションと実環境のギャップによって，進化したロボットが実環境で適切に振舞えないという問題が生じる [5]．この問題に対しては多くのアプローチ [4][6][7] が提案されているが，シミュレーションを用いることができる問題は，その環境およびロボットとの相互作用が適切にモデル化できる問題に限られる．そこで本稿では，実環境で実ロボットを評価することに焦点を当てる．

では，どのようにすれば実環境でロボットを進化させる時間を短縮できるであろうか．この問題に対し，人工進化が解くべき問題の探索空間をできるだけ小さく設定することを考える．つまり，遺伝子長を短くすることができれば，どの進化アルゴリズムを採用しても基本的には進化時間を短縮することが可能であると期待できる．本稿では，ロボットの制御器として人工ニューラルネットワーク (Artificial Neural Networks: ANN) を採用し，遺伝アルゴリズムによって設計を行う．そして，遺伝子長が短くなるように，遺伝子コーディングとして考えうる最も簡単なものを採用する．この簡易コーディングを用いた ANN を移動ロボットに搭載し，計算機において進化実験を行い，その性能を検証する．

## 2. 簡易コーディングを用いた進化型ニューラルコントローラ

本節では，簡易コーディングを用いた進化型ニューラルコントローラの概略を示す．

ロボットの制御器として，中間層・出力層が相互結合された ANN を用いる．$i$ 番目のニューロンの時刻 $n$ での出力は以下の式で表される．

$$x_i(n) = f(\sum_{j=1}^{m} \omega_{ij} x_j(n-1)) \tag{1}$$

ここで，$\omega_{ij}$ は $j$ 番目のニューロンから $i$ 番目のニューロンへの結合荷重であり，$f(x)$ はニューロンの出力関数であり，式 (2) で与えられるシグモイド関数を用いる．

$$f(x) = \frac{1}{1 + exp(-x/T)} \tag{2}$$

$T$ はシグモイド関数の傾きを決める定数である．一般的に進化型 ANN ではその変数として，各ニューロン間の結合荷重・ニューロンのしきい値 (シグモイド関数を用いる場合はその傾き)・ニューラルネットの構造

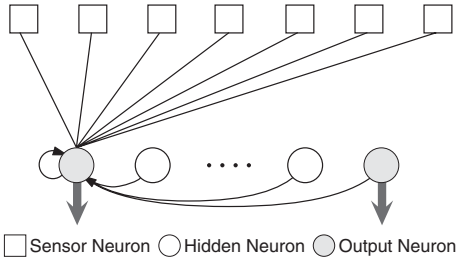



**Fig.1** Architecture of ANNs for 1 block of a string

が挙げられるが，本稿では遺伝子長を短くするために結合荷重のみを GA の変数とする．シグモイド関数の傾きは $T = 1$ とする．ストリングへの $\omega$ のコーディング法として以下の 3 つを用いる．

**Simple ANN 1 (SANN1)**

Floreano ら [8] は実ロボットの進化に次のようなコーディング法を用いている．ストリングは $n$ 個のブロックから構成され，1 個のブロックは中間層･出力層を合わせた全ノードのそれぞれに定義される (図 1)．ブロックの最初のビットはそのノードに入る全結合の符号 $\{+,-\}$ を表し，残りのビットは $N_s$ 個の入力ノードおよび $n$ 個の中間・出力ノードからの結合の有無 $\{1,0\}$ を表す (図 2(a))．中間ノード数を $N_h$，出力ノード数を $N_o$ とすると，遺伝子長は $L = (N_h+N_o)(1+N_s+N_h+N_o)$ ビットとなる．

**Simple ANN 2 (SANN2)**

Floreano らのコーディング法 (SANN1) を拡張したものとして，中間層・出力層のノードに入る結合それぞれに符号を設定する．つまり，ブロックはそのノードに入る結合の符号 $\{+,-\}$ および結合の有無 $\{1,0\}$ で構成される (図 2(b))．SANN1 よりも ANN の構造に多様性を持たせられる．遺伝子長は $L = 2(N_h + N_o)(N_s + N_h + N_o)$ ビットとなる．

**Simple ANN 3 (SANN3)**

SANN1･SANN2 で考えた結合の有無をコーディングせず，入力層・中間層・出力層のノードを全結合とする．つまり，ブロックはそのノードに入る結合の符号 $\{+,-\}$ のみで構成される (図 2(c))．遺伝子長は $L = (N_h + N_o)(N_s + N_h + N_o)$ ビットとなる．

## 3. 計算機実験

### 3.1. 設定

本節では進化ロボティクスの 1 問題である目標物到達問題において 2. で示した進化型ニューラルコントローラーの性能を検証する．図 3 に目標物到達問題の概略を示す．ロボットは近接センサを搭載しており，図に示す距離まで測定が可能であり，物体までの距離と反比例する値を出力するように設定する．ロボットの移動機構を左右二輪独立駆動型と仮定し，式 (3) の

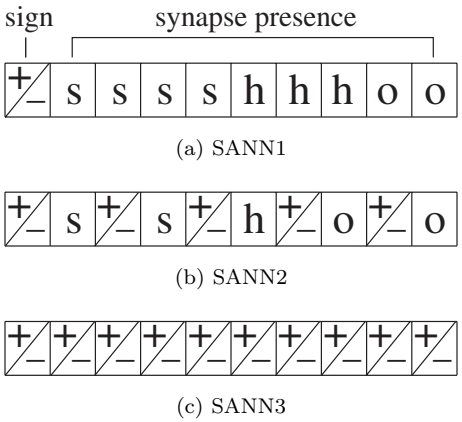



**Fig.2** Genetic representation of one block

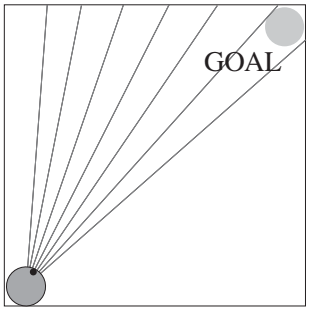



**Fig.3** Experimental setup for a goal reach problem

ようにモデル化する (図 4)．

$$\begin{aligned} x_{k+1} &= x_k + \frac{V_R + V_L}{2}\cos\theta_k + \omega_{xk+1} \\ y_{k+1} &= y_k + \frac{V_R + V_L}{2}\sin\theta_k + \omega_{yk+1} \\ \theta_{k+1} &= \theta_k + \frac{V_R - V_L}{2R} + \omega_{\theta k+1}, \end{aligned} \tag{3}$$

ここで，$V_R$ と $V_L$ は両車輪の周速度，$2R$ は車輪間隔，$\omega_{[\cdot]k}$ はシステム誤差のモデルとする．式 (3) 中のシステム誤差は平均 0・標準偏差 $\sigma$ の正規分布 $N(0,\sigma)$ で表されることが知られている [9]．本実験では，$\sigma$ は各ステップにおける移動量の 10%に設定する．

ロボットの制御器として，2. で示した ANN を用いる．入力ノード数は $N_s = \{1,2,3,5,7\}$(近接センサ数)，中間ノード数は $N_h = \{1,3,5\}$ と変化させる．出

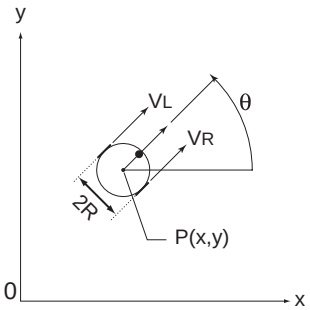



**Fig.4** Simulated model for a mobile robot

力ノード数は $N_o = 2$(左右モータ) とする．一般的に入力ノード数は解くべき問題やロボットの設定 (センサ数) に依存するので，本実験では広い範囲で変化させて ANN の性能を検証する．一方，中間ノード数は ANN の設計者が問題ごとに決定するが，1. で述べたように遺伝子長を短くすることが目的であるため，比較的小さい数で実験を行う．

ロボットは，より早く目標物に到達することを目的とする．ロボットの初期姿勢を変えた試行を 8 回行い，次の適応度関数を用いて適応度を計算する:

$$Fitness = \frac{1}{NumTrials} \sum_{i=1}^{NumTrials} (1 - \frac{Step}{MaxStep}) \tag{4}$$

ここで，$NumTrials$ は試行回数 (8) であり，$MaxStep$ は 200 とする．

進化手法として Simple GA(以下 SGA) および ANN を設計する問題に対しその有効性が確認されている Operon GA(以下 OGA[10]) を用いる [11][12]．ANN の結合荷重を変数とし，2. で述べたようにコーディングを行う．文献 [11] と同様の理由により，交叉は両 GA ともに用いない．SGA では遺伝的操作として点突然変異のみを用いる．両 GA ともに点突然変異率を $q = 1/L$ とする．OGA では遺伝的操作として点突然変異に加えて 5 つの遺伝的操作を用いる．これらは結合，分割，重複，欠失，逆位とよばれ，各パラメータは文献 [11] における推奨値を参考に，$(g_{con}, g_{div}, g_{dup}, g_{del}, g_{inv}) = (0.3, 0.3, 0.6, 0.3, 0.3)$ を使用する．各遺伝子座における値のリスト長は 6 とする．両 GA ともに，選択法としてトーナメント選択を用い，エリート戦略[1] を適用する．SGA では弱い選択圧が，OGA では強い選択圧が推奨されていることから，SGA ではトーナメントサイズを 2 に，OGA では 6 に設定する．個体数を 50，世代数を 1000 とし，各計算回数を 10 回とする．

## 3.2. 実験結果と考察

### 3.2.1 最終世代における適応度

図 5 に各コーディング法を用いた場合の最終世代における入力ノード数に対する各 GA の最大適応度の平均を示す．どのコーディング法においても，入力ノード数が増加すると適応度は増加する傾向にあることがわかる．中間ノード数でみると，SANN1 では，$N_h = 3$ のときがおおむね良いパフォーマンスを示している．また，SANN2 および SANN3 では $N_h = 3, 5$ のときに良いパフォーマンスを示している．コーディング法で比べると，同一の入力・中間ノード数では SANN2 および SANN3(ただし，$N_h = 1$ のときを除く) が SANN1 よりも良いパフォーマンスを示している．進化アルゴリズムで比較すると，多くの場合 OGA が SGA よりも良いパフォーマンスを示している．

[1] 適応度が最も高い個体からランダムに 1 つの個体を次世代の親個体として引き継ぐ.

### 3.2.2 適応度の推移

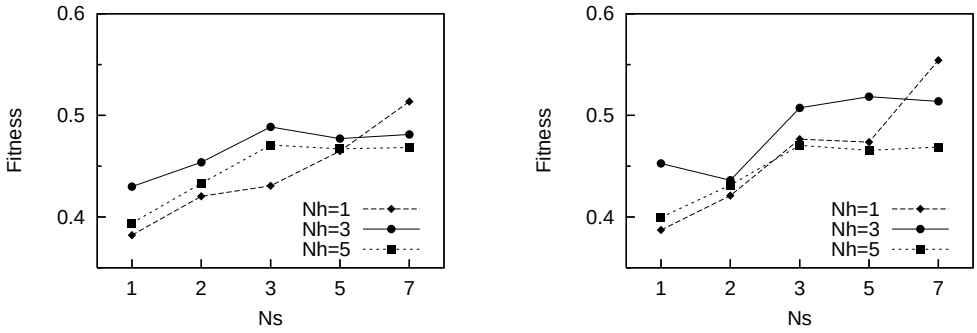
次に，最終世代で良いパフォーマンスを示した SANN2 と SANN3 の $N_h = 3, 5$ に関して，世代数に対する各 GA の最大適応度の推移を図 6-9 に示す．SGA を用いた場合，SANN2 の $N_h = 3, 5$(図 6) および SANN3 の $N_h = 3, 5$(図 8) において一定期間，適応度が変化しない equilibrium period[13] を多く含んでいることが確認できる．これにより高い適応度を得るまでに世代数を要している．また，$N_h = 5$ よりも $N_h = 3$ の方が早く適応度が高くなっている．一方，OGA を用いた場合 equilibrium period はほとんど見られない (図 7,9)．$N_h = 3$ ではほぼ 200 世代までに高い適応度を獲得している (図 7(a),9(a))．$N_h = 5$ でも早い世代数で高い適応度を獲得しているが $N_h = 3$ と比較すると高い適応度を得るまでに世代数が多くかかっている (図 7(b),9(b))．先に述べたように，最終世代における適応度で比較すると SANN2 および SANN3 は同様に $N_h = 3, 5$ のときに良いパフォーマンスを示しているが，高い適応度を得るまでの世代数で比較すると $N_h = 3$ の方がより早く適応度が高くなっている．この結果から，OGA を用いる SANN2・SANN3(ともに $N_h = 3$) が 1. の目的に対して望ましいと考えられる．

### 3.2.3 獲得した ANN の構造

図 10-12 に $(N_s, N_h)$ の各組み合わせの中で最も高い適応度を獲得した ANN の構造を各コーディング法ごとに示す．SANN1 では OGA を用いた $(N_s, N_h) = (7, 1)$ が最も高い適応度を獲得し，非常にシンプルな構造となっている (図 10)．各中間・出力ノードに入る全結合荷重値の符号がストリング中の 1 ビットで決定されているため，ロボットの振舞い解析を行う際にはそのノードが振舞いに与える影響を調べやすいということも期待できる．SANN2 では OGA による $(N_s, N_h) = (7, 5)$ が最も高い適応度を獲得した．結合荷重値の符号が各結合ごとにコーディングされているため，ANN の構造に多様性を持たせることができている (図 11)．このことは SANN2 が各 $(N_s, N_h)$ の組み合わせにおいても良いパフォーマンスを示した要因であると考えられる．SANN3 では，SGA による $(N_s, N_h) = (7, 3)$ が最も高い適応度を獲得した．SANN3 は入力層・中間層・出力層のノードが全結合しており，結合荷重値の符号のみをコーディングしている．このことは，SANN2 と同様に ANN の構造の多様性につながっている (図 12)．そのため，各ノードはエージェントの振舞いにそれぞれ影響を与えることとなる．

### 3.2.4 最良個体による振舞い

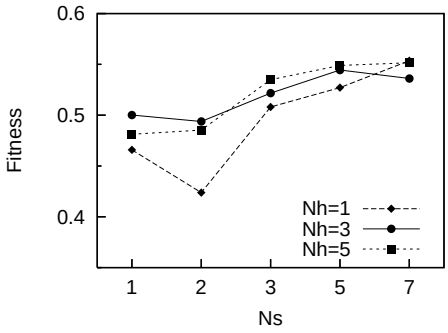
図 13 に本実験において，最終世代で最も高い適応度を獲得した，SGA による SANN3($(N_s, N_h) = (7, 3)$) のコントローラ (図 12) を用いたロボットの振舞いを示す．初期位置において目標物方向正面になるまで右



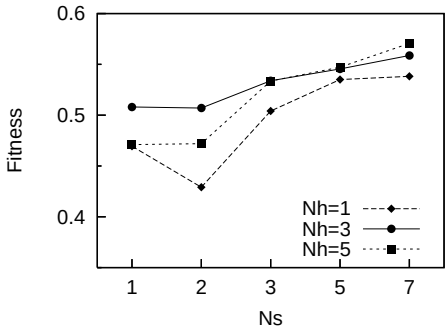

(a) SANN1 by the SGA



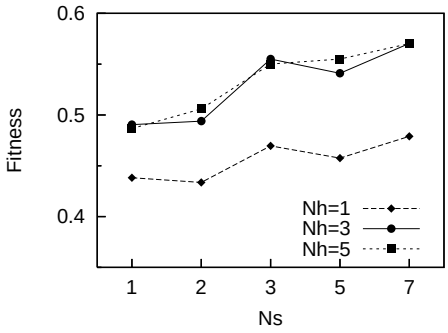

(b) SANN1 by the OGA



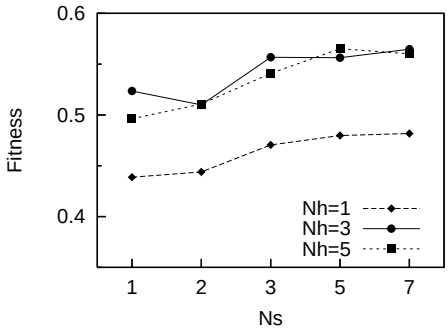

(c) SANN2 by the SGA




(d) SANN2 by the OGA




(e) SANN3 by the SGA




(f) SANN3 by the OGA

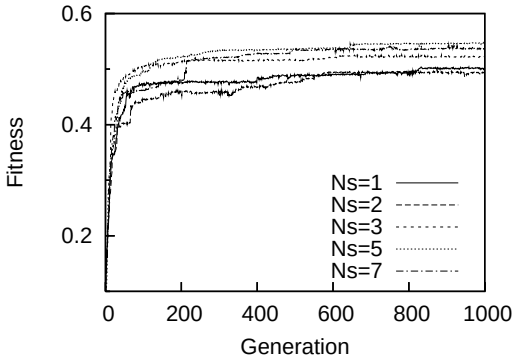
**Fig.5** Maximum fitness at last generation for each coding by the SGA and the OGA

回転し，その後一直線に目標物に向かっている．本実験で採用した適応度関数 (式 (4)) はより早く目標物に到達するというものであった．したがって，センサー入力が入るまでは回転以外の無駄な行動はせず，センサー入力が入るとモーターに接続されている出力ノードが活性化し，直線運動を行い目標物に到達していると考えられる．このことから，簡易コーディングを用いた進化型ニューラルコントローラをもつロボットは望ましい振舞いを獲得できることを確認した．

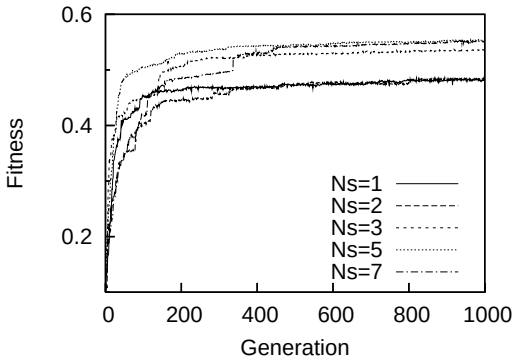
## 4. おわりに

実ロボットを実環境で進化させるには膨大な時間がかかる．本稿では，この実時間を短縮するために，人工進化が解くべき問題の探索空間をできるだけ小さくすることを目的として，進化型ニューラルコントローラの簡易コーディング法を提案した．そして，進化ロボティクスの 1 問題である目標物到達問題において，2 種類の遺伝アルゴリズムを適用し，その性能を検証した．得られた結果について次にまとめる．

- どのコーディング法においても，入力ノード数が増加するとパフォーマンスは良くなる．
- SANN1 では $N_h = 3$ のとき，SANN2 および SANN3 では $N_h = 3,5$ のときに良いパフォーマンスを示す．
- 同一の入力・中間ノード数で比較した場合，SANN2 および SANN3 が SANN1 よりも良いパフォーマンスを示す (ただし，$N_h = 1$ のときを除く)．
- 多くの場合に OGA を適用した ANN は SGA のそれよりも良いパフォーマンスを示す．
- SANN2 および SANN3 において $N_h = 3$ と $N_h = 5$ を比較した場合，$N_h = 3$ の方がより早い世代で高い適応度を獲得する．

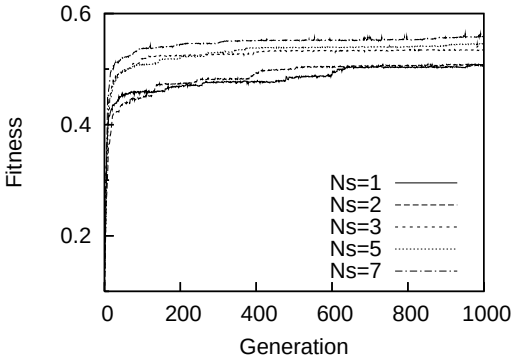



(a) $N_h = 3$

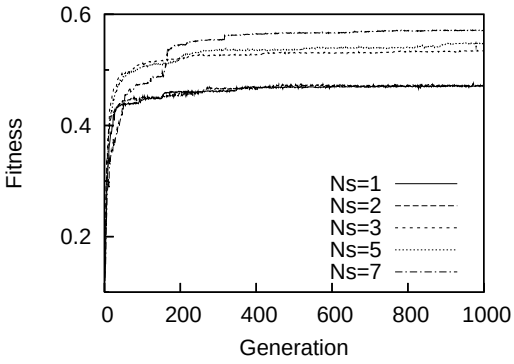



(b) $N_h = 5$

**Fig.6** Maximum fitness at each generation for SANN2 by the SGA

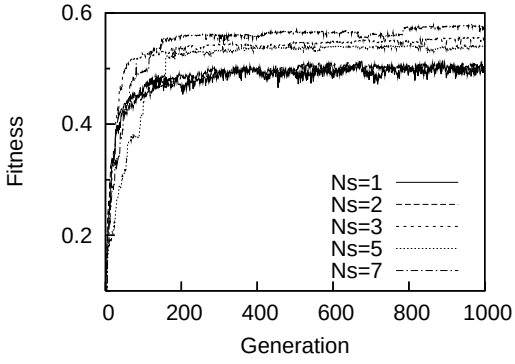



(a) $N_h = 3$

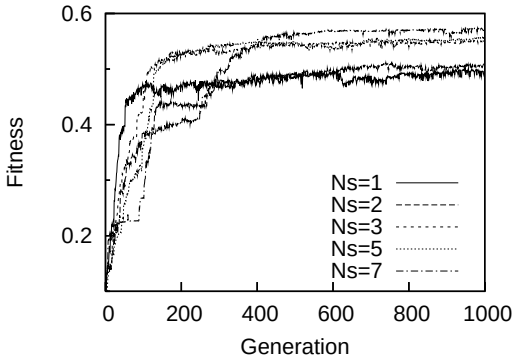



(b) $N_h = 5$

**Fig.7** Maximum fitness at each generation for SANN2 by the OGA

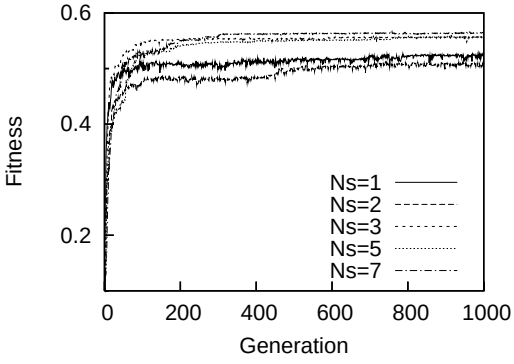



(a) $N_h = 3$

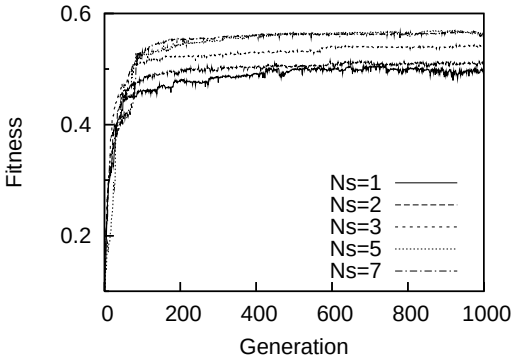



(b) $N_h = 5$

**Fig.8** Maximum fitness at each generation for SANN3 by the SGA




(a) $N_h = 3$




(b) $N_h = 5$

**Fig.9** Maximum fitness at each generation for SANN3 by the OGA

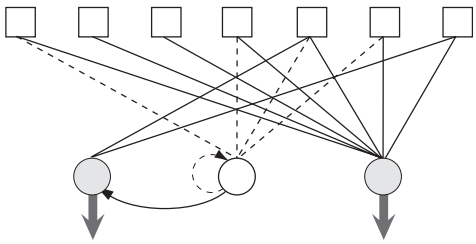


**Fig.10** A genetically determined controller for SANN1 with $(N_s, N_h) = (7, 1)$ by the OGA

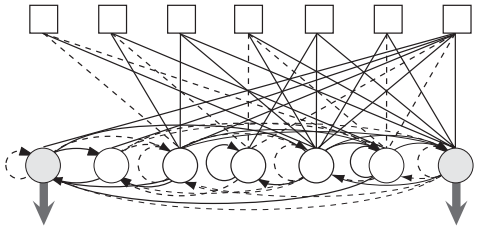


**Fig.11** A genetically determined controller for SANN2 with $(N_s, N_h) = (7, 5)$ by the OGA

以上より，$N_h = 3$ をもつSANN2・SANN3 にOGA を適用することがパフォーマンスおよび進化速度の観点から有効であることを確認した．遺伝子長を考えた場合，同じ $(N_s, N_h, N_o)$ の組み合わせでSANN2 の遺伝子長はSANN3 のそれの 2 倍になる (2. 参照)．遺伝子長を短く抑えるということを考慮するとSANN3 がより適切であると考えられる．今後，本稿で得られた知見をもとに，実環境における進化ロボティクスの問題や協調型共進化の問題に挑戦したい．

## 参考文献


[1] R. Pfeifer and C. Scheier, Understanding Intelligence, The MIT Press, 1999.

[2] Stefano Nolfi and Dario Floreano: Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines, MIT Press, (2000)

[3] R. A. Watson, S. G. Ficici and J. B. Pollack, Embodied Evolution: Embodying an Evolutionary Algorithm in a Population of Robots, In Proceedings of Congress on Evolutionary Computation, pp.335-342, (1999)

[4] N. Jakobi, Half-baked Ad-hoc and Noisy: Minimal Simulation for Evolutionary Robotics, In Proceedings of the Fourth European Conference on Artificial Life, pp.348-357, 1997.

[5] R. A. Brooks, Artificial Life and Real Robots, In Proceedings of the First European Conference on Artificial Life, pp.3-10, 1992.

[6] O. Miglino, H. H. Lund and D. Nolfi, Evolving Mobile Robots in Simulated and Real Environments, Artificial Life 2, pp.417-434, 1995.

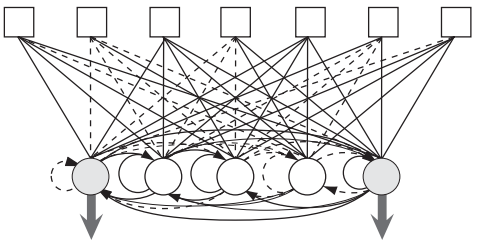



**Fig.12** A genetically determined controller for SANN3 with $(N_s, N_h) = (7, 3)$ by the SGA

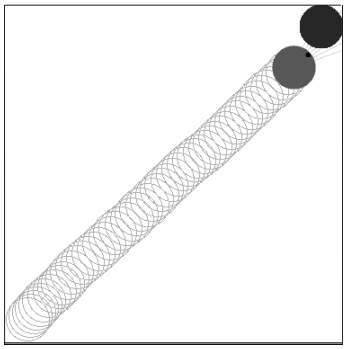


**Fig.13** Behavior of the genetically determined controller for SANN3 with $(N_s, N_h) = (7, 3)$ by the SGA


[7] 高崎, 片田, 大倉, 田浦: 進化型自律移動ロボットのオンラインモデル更新に関する一手法, 第 47 回自動制御連合講演会, CD-ROM 予稿集, 404, (2004)

[8] D. Floreano and C. Mattiussi: Evolution of Spiking Neural Controllers, Proc. of Evolutinary Robotics -From Intelligent Robots to Artifical Life (ER'01), pp.38-61, Springer-Verlag, (2001)

[9] 小森谷, 大山, 谷: 移動ロボットのためのランドマーク観測計画, 日本ロボット学会誌, Vol.11, No.4, pp.533-540, (1993)

[10] 大倉，上田: 中立突然変異型 GA による騙し問題の最適化, 計測自動制御学会論文集, Vol. 32, No. 10, pp. 1461-1469, (1996)

[11] 片田，大倉，上田: Neutral Networks を含む適応度景観における遺伝的アルゴリズムの進化ダイナミクス, システム制御情報学会論文誌, Vol. 17, No. 5, pp. 187-195, (2004)

[12] Y. Katada, K. Ohkura and K. Ueda: An Approach to Evolutionary Robotics Using the Genetic Algorithm with Variable Mutation Rate Strategy, Proc. of Parallel Problem Solving from Nature (PPSN VIII), pp. 952-961, (2004)

[13] E. Nimwegen, J. Crutchfield and M. Mitchell: Statistical Dynamics of the Royal Road Genetic Algorithm, Theoretical Computer Science, Vol. 229, No. 1, pp. 41-102, (1999)