論　文

強化学習による4足ロボットの歩行動作獲得

正　員　木　村　　元（東京工業大学）
非会員　山　下　　透（東京工業大学）
非会員　小　林　重　信（東京工業大学）

Reinforcement learning of walking behavior for a four-legged robot

Hajime Kimura, Member, Toru Yamashita, Non-member, Shigenobu Kobayashi, Non-member (Tokyo Institute of Technology)

We investigate a reinforcement learning of walking behavior for a four-legged robot. The robot has two servo motors per leg, so this problem has eight-dimensional continuous state/action space. We present an action selection scheme for actor-critic algorithms, in which the actor selects a continuous action from its bounded action space by using the normal distribution. The experimental results show the robot successfully learns to walk in practical learning steps.

キーワード：強化学習，ロボット，行動選択，ガウス分布
Keywords: reinforcement learning, actor-critic, robotics, exploration strategy, normal distribution,

1. はじめに

強化学習は，報酬という単純な指示から，それよりもはるかに複雑な制御規則を自動的に獲得するための手法として有望である[1]．強化学習で最も一般的なのはQ-learning[2]のように状態や行動の評価値(value)を推定した後に最適な政策を得る価値関数アプローチ(value-function approach)と呼ばれる方法である．実環境におけるロボットの学習制御においては連続で膨大な状態および行動空間の扱いが求められる．連続な状態または行動空間において価値関数(value function)を近似する方法については多くの研究例[3]がある．空間をグリッドで分割する方法が最も単純であり，TD法[4]やQ-learningの収束が保証される線形アーキテクチャ(linear architecture)[5][6]の性質を有する関数近似方法の一つである．しかし，分割が粗いと非マルコフ問題が発生し，細かすぎると汎化性能の低下という問題が生じる．CMAC[4]はグリッド分割を拡張することで線形アーキテクチャの性質を保ったまま上記の問題点の解決を図る関数近似方法として知られる．訪問頻度の高い状態領域での関数近似精度を高める工夫として，事例に基づく方法(instance-based method)[7][8]や適応的状態分割[9],[10]などの方法が提案されている．しかし，状態空間と行動空間が高次元の場合，最適政策を求めるために状態-行動評価値(state-action value)を正確に推定することは，推定に要するサンプルを得る試行回数が膨大であり，その関数近似のためには膨大なリソース(メモリ)を必要とする．よってマルコフ決定過程の環境下での強化学習において一般的な方法論である状態-行動評価関数を推定してから最適政策を得る価値関数アプローチは実行不能である．

Actor-criticアルゴリズムは，政策勾配法(policy gradient methods)に基づく別の接近法である．政策勾配法は，確率的政策のパラメータを価値関数の勾配を用いて更新していく方法である．このアプローチは局所解に陥る可能性を伴うものの，多くの利点を有するため，高次元の空間を持つ強化学習の実問題においてたびたび用いられている[11][12]．本論文ではactor-criticアルゴリズムを4足ロボットの歩行動作獲得問題へ適用するための新しい実装方法を提案する．ロボットは1脚あたりモータ2個を持つため，全体で8次元の連続な状態空間と同時に8次元連続値の行動空間を持つ強化学習問題である．このような問題においてactor-criticを実装する場合，actorの行動選択に正規分布を用いる方法が多いが，正規分布は定義域に上界／下界が無い．そのため環境で定義されている行動空間の外の行動を選択する場合には行動空間の上界または下界の値を指示したものとして処理するが，これは特に本論文のロボットの場合には学習途中で全く動かなくなるなどの悪影響が生じるという問題がある．そこで本論文では，連続な行動空間に上下界の存在する環境において，正規分布を用いて行動選択を行うための新しい実装方法を提案する．実機を用いた実験では，ロボットは提案手法により現実的な時間内で歩行動作を獲得することに成功した．

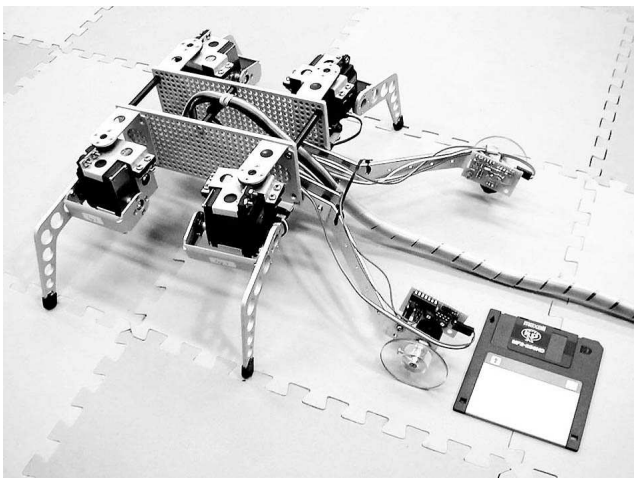

図 1　使用したロボット．各足は角度指令に基づいて動作するサーボモータ 2 個を持つ．

Fig. 1. The robot. Each leg is controlled by 2-servo motors that react to angular-position commands.

2. 問題設定

〈2･1〉 4 足ロボット　本論文では Fig.1 に示す 4 足ロボットを扱う．学習目標は，なるべく速くまっすぐに前進する制御規則を獲得することである．このロボットを制御するコントローラが学習主体の「エージェント」になり，ロボット本体を含めた外界がエージェントにとっての「環境」になる．エージェントは事前に環境やロボットのダイナミクスを知らされていないため，試行錯誤を通じて制御規則を形成していくことが求められる．エージェントの保持する制御規則は，状態入力から行動への確率分布という形式の政策関数で表現される．エージェントは試行錯誤を通じて政策関数のパラメータを改善する．ロボットの各脚は Fig.1 に示すように 2 個の位置制御サーボモータで駆動され，全体で 8 自由度である．各脚の 2 つのモータはそれぞれリフト脚，スイング脚を受け持つ．リフト脚は上下方向，スイング脚は前後方向へ足先を運ぶ役割を持つ．

各時間ステップにおいてエージェントは 8 個のサーボモータの角度を観測して現在の状態とし，政策関数に従って行動を選択する．行動は 8 個のサーボモータの角度の目標値である．よって現在の状態は 1 ステップ前に出力した行動に等しい．行動を実行後，約 0.3 秒後に状態遷移結果として報酬が与えられ，次の時間ステップへと進む．Fig.1 の右側に示される 2 個の車輪は，ボディの移動を検出して報酬信号を生成する．2 つの車輪の移動速度平均はロボットの前進速度を示し，2 つの車輪の移動速度の差分はロボットが旋回していることを示す．ロボットの学習目標はまっすぐに前進することなので，各時間ステップでの報酬は，2 つの車輪の移動速度平均から差分の絶対値を引いた値とした．車輪は直径 5cm で 200 パルス / 回転の信号を生成する．

〈2･2〉 マルコフ決定過程によるモデル化　本論文ではロボットの学習問題をマルコフ決定過程 (MDP) における強化学習問題として以下のように定式化する．状態空間を $\mathcal{S}$，行動空間を $\mathcal{A}$，実数の集合を $\mathcal{R}$ と表す．各離散時間ステップ t において，エージェントは状態 $s_t \in \mathcal{S}$ を観測して行動 $a_t \in \mathcal{A}$ を実行し，状態遷移の結果，報酬 $r_t \in \mathcal{R}$ を受け取る．一般に報酬と遷移先の状態は確率変数だが，MDP ではその分布は s_t と a_t だけに依存する．遷移先の状態 s_{t+1} は遷移確率 $T(s_t, a, s_{t+1})$ に従い，報酬 r_t は期待値 $r(s_t, a)$ に従って与えられる．

学習エージェントは $T(s_t, a, s_{t+1})$ や $r(s_t, a)$ についての知識は事前に与えられない．強化学習の目的はエージェントのパフォーマンスを最大化する政策 (policy) を生成することである．無限期間のタスクにおいて自然な評価規範としては，以下のような割引報酬合計による評価がある．

$$V_t = \sum_{k=0}^{\infty} \gamma^k \, r_{t+k} \ , \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \ (1)$$

ただし割引率 $0 \leq \gamma \leq 1$ は未来の報酬の重要度を示し，V_t は時刻 t の評価値を表す．MDP では，評価値は以下のように定義できる：

$$V^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t \, r_t \middle| s_0 = s, \pi \right] \ , \quad \cdots\cdots\cdots \ (2)$$

ただし $E\{\cdot\}$ は期待値演算を表す．MDP では式 2 で定義される各状態 s における評価値を最大化する政策を探すことが目標である．本論文のロボット学習問題では，状態空間が上下界の存在する 8 次元の連続な空間であり，行動空間も同じである．

3. 強化学習アルゴリズム

〈3･1〉 **Actor-Critic** アルゴリズム　Actor-critic アーキテクチャ [13] (Fig.2) は MDP だけでなく非マルコフ問題の一種である POMDP におけるあるクラスにおいても有望な手法である．Actor は，状態から行動への確率分布である" 確率的政策 (**stochastic policy**)" に従って行動を実行する．Critic は，actor の政策のもとでのそれぞれの状態の" 価値 (**value**)" を推定する．Actor は，critic で計算される *temporal difference (TD)* エラーを手がかりにして政策を改善する．政策の評価値推定のプロセスの収束を待って政策改善を行うのが理想的だが，多くの場合，政策改善と政策の評価値推定のプロセスは同時に行われる．

Fig.3 は本ロボットに適用した actor-critic の詳細である．Actor と critic の両方に適正度の履歴 (eligibility trace) [14] を用いているのが特徴である．Critic では TD(λ) でそれを用いており，actor では政策パラメータについて用いている [15] [16] ．

〈3･1･1〉 **Critic** における処理手順　Critic における TD(λ) の処理は Fig.3 のステップ 3 と 5 の部分である．パラメータ λ_v は適正度の履歴を特徴付け，TD($\lambda = \lambda_v$) となる．

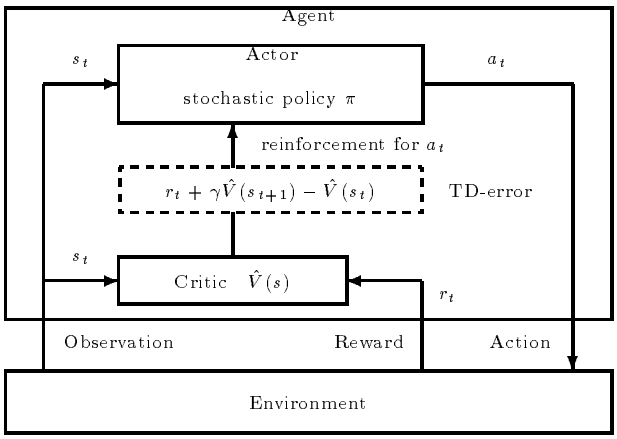

図 2　一般的な actor-critic アルゴリズムの枠組．Critic は状態評価値を推定し actor に TD-error を与える．Actor はそれに基づいて政策を改善する．TD-error > 0 の場合，行動 a_t は良い行動と考えられるので a_t をとる確率を上げる．さもなければ a_t の確率を下げる．

Fig. 2. A standard actor-critic architecture. The critic estimates state values and provides the TD-error to the actor. The actor updates the policy using it. If the TD-error > 0, the actor raises probability of action a_t because the action a_t would lead the agent to a better state. Otherwise, it decreases the probability of a_t.

〈3･1･2〉 **Actor における処理手順**　Actor では，確率的政策はパラメータ関数表現され，そのパラメータを価値関数の勾配を用いて更新していく [16] [17]．エージェントが政策 π のもとで観測 s において行動 a を選択する確率を関数 $\pi(a|\theta, s)$ で表す．政策 $\pi(a|\theta, s)$ は行動 a の集合が連続値の場合は確率密度関数である．エージェントは内部変数 θ を調節することにより確率的政策 π を変える．Actor の更新処理は Fig.3 のステップ 4 と 5 である．パラメータ λ_π は actor の適正度の履歴を特徴付けるが，TD(λ) の場合とは性質がやや異なる．λ_π が 0 に近い場合，critic で推定された価値関数 $\hat{V}$ を用いて政策が更新される．λ_π が 1 に近い場合，式 1 で定義される実際の報酬のサンプル合計による評価値を用いて政策が更新される．

〈3･2〉 **4 足ロボット学習問題のための実装**　ロボットの 8 個の関節モータ角度をベクトル $(s_1, s_2, \cdots, s_8)$ によって表現し，これを状態とする．ただし各要素は $-1 \leq s_i \leq 1$，$i = 1, 2, \cdots, 8$ のように正規化されているものとする．Critic では 8 次元の連続な状態空間は各座標軸毎に 2 分割，合計 2^8 個の超矩形領域によって離散化され，状態は 2^8 の長さを持つ単位ベクトル $(x_1, x_2, \cdots, x_{256})$ を用いて，該当する領域の要素のみ 1，それ以外の要素は 0 という具合に表現される．Fig.3 中で表される推定値 $\hat{V}(s_t)$ は以下のように計算される：

$$\hat{V}(s_t) = \sum_{b=1}^{256} x_b w_b \ , \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (6)$$

(1) Observe state s_t, choose action a_t with probability $\pi(a_t|\theta, s_t)$, and perform it.

(2) Observe immediate reward r_t, resulting state s_{t+1}, and calculate the TD-error according to

$$\text{(TD-error)} = r_t + \gamma\hat{V}(s_{t+1}) - \hat{V}(s_t) \ , \quad \cdots\cdots (3)$$

where $0 \leq \gamma \leq 1$ is the discount factor, $\hat{V}(s)$ is an estimated value function by the critic.

(3) Update the estimating value function $\hat{V}(s)$ in the critic according to the TD(λ) method as:

$$e_v(t) = \frac{\partial}{\partial w}\hat{V}(s_t) \ , \quad \cdots\cdots\cdots\cdots\cdots\cdots (4)$$
$$\overline{e_v}(t) \leftarrow e_v(t) + \overline{e_v}(t) \ ,$$
$$\Delta w(t) = \text{(TD-error)}\,\overline{e_v}(t) \ ,$$
$$w \leftarrow w + \alpha_v \Delta w(t) \ ,$$

where e_v denotes the eligibility of the parameter w in the function approximator $\hat{V}(s)$, $\overline{e_v}$ is its trace, and α_v is a learning rate.

(4) Update the actor's stochastic policy by

$$e_\pi(t) = \frac{\partial}{\partial\theta}\ln\Big(\pi(a_t|\theta, s_t)\Big) \ , \quad \cdots\cdots\cdots\cdots (5)$$
$$\overline{e_\pi}(t) \leftarrow e_\pi(t) + \overline{e_\pi}(t) \ ,$$
$$\Delta\theta(t) = \text{(TD-error)}\,\overline{e_\pi}(t) \ ,$$
$$\theta \leftarrow \theta + \alpha_\pi \Delta\theta(t) \ ,$$

where e_π is the eligibility of the policy parameter θ, $\overline{e_\pi}$ is its trace, and α_π is a learning rate.

(5) Discount the eligibility traces as follows:

$$\overline{e_v}(t+1) \leftarrow \gamma\lambda_v\,\overline{e_v}(t) \ ,$$
$$\overline{e_\pi}(t+1) \leftarrow \gamma\lambda_\pi\,\overline{e_\pi}(t) \ ,$$

where λ_v and λ_π $(0 \leq \lambda_v, \lambda_\pi \leq 1)$ are discount factors in the critic and the actor respectively.

(6) Let $t \leftarrow t + 1$, and go to step 1.

図 3　適正度の履歴を用いた actor-critic アルゴリズム

Fig. 3. An actor-critic algorithm using eligibility traces in both the actor and the critic.

ただし w_b は関数近似のためのパラメータである．式 4 の $e_v(t)_b$ は b^{th} 番目のパラメータ w_b の適正度 (eligibility) である．すなわち $e_v(t) = (e_v(t)_1, e_v(t)_2, \cdots e_v(t)_b \cdots, e_v(t)_{256})$，このとき式 6 より

$$e_v(t)_b = \begin{cases} 1 & \text{, where } x_b = 1 \ , \\ 0 & \text{, where } x_b = 0 \ . \end{cases} \quad \cdots\cdots\cdots\cdots (7)$$

ロボットの 8 個の関節モータ角度の目標値をベクトル $(a_1, a_2, \cdots, a_8)$ によって表現し，これを行動とする．ただし各要素は状態ベクトルと同様に $-1 \leq a_i \leq 1$，$i = 1, 2, \cdots, 8$ のように正規化されているものとする．Actor は以下のような行動選択を行う：各モータ (i) において，actor は正規分布 $N(\mu_{(i)}, \sigma^2_{(i)})$ に従い，$[-1, 1]$ の範囲の値を得るまでランダムサンプルをとり続け，その値が出たら

それを選択する．正規分布のパラメータ $\mu_{(i)}$ と $\sigma_{(i)}$ は以下のシグモイド関数によって与える：

$$\mu_{(i)} = \frac{2}{1+\exp\left(-\sum_{k=1}^{8} s_k \theta_{k,(i)}\right)} - 1 \ ,$$

$$\sigma_{(i)} = \frac{1}{1+\exp\left(-\theta_{9,(i)}\right)} \ ,$$

ただし $\theta_{k,(i)}$ $(k = 1, 2, \cdots 9)$ は政策パラメータである．正規分布 $N(\mu_{(i)}, \sigma^2_{(i)})$ が 1 回の試行で $[-1, 1]$ の範囲のサンプルを生成する確率を P_{in} とすると，

$$P_{in} = \int_{-1}^{1} \frac{1}{\sigma_{(i)}\sqrt{2\pi}} \exp\left(-\frac{(x-\mu_{(i)})^2}{2\sigma^2_{(i)}}\right) dx \quad \cdot \ (8)$$

このとき政策関数は以下で表される：

$$\begin{aligned}
&\pi(a_{(i)}|\theta, s_t\ ) \\
&= \left(1 + (1-P_{in}) + (1-P_{in})^2 + \cdots\right) \\
&\quad \times \frac{1}{\sigma_{(i)}\sqrt{2\pi}} \exp\left(-\frac{(a_{(i)}-\mu_{(i)})^2}{2\sigma^2_{(i)}}\right) \\
&= \frac{1}{P_{in}} \frac{1}{\sigma_{(i)}\sqrt{2\pi}} \exp\left(-\frac{(a_{(i)}-\mu_{(i)})^2}{2\sigma^2_{(i)}}\right) \ , \quad \cdots \ (9)
\end{aligned}$$

ただし $a_{(i)}$ は $[-1, 1]$ の範囲に制限される．i^{th} 番目のモータにおける k^{th} 番目の政策パラメータ $\theta_{k,(i)}$ の適正度を $e_\pi(t)_{k,(i)}$ と表すと，式 5 中の θ と $e_\pi(t)$ は 9×8 のマトリクスによって表現される：

$$\theta = \begin{Bmatrix} \theta_{1,(1)} & \theta_{2,(1)} & \cdots & \theta_{9,(1)} \\ \theta_{1,(2)} & \theta_{2,(2)} & \cdots & \theta_{9,(2)} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{1,(8)} & \theta_{2,(8)} & \cdots & \theta_{9,(8)} \end{Bmatrix}$$

$$e_\pi(t) = \begin{Bmatrix} e_\pi(t)_{1,(1)} & e_\pi(t)_{2,(1)} & \cdots & e_\pi(t)_{9,(1)} \\ e_\pi(t)_{1,(2)} & e_\pi(t)_{2,(2)} & \cdots & e_\pi(t)_{9,(2)} \\ \vdots & \vdots & \ddots & \vdots \\ e_\pi(t)_{1,(8)} & e_\pi(t)_{2,(8)} & \cdots & e_\pi(t)_{9,(8)} \end{Bmatrix}$$

式 5，9 より各 $e_\pi(t)_{k,(i)}$ は以下で与えられる．

$$\begin{aligned}
e_\pi(t)_{k,(i)} &= \frac{\partial}{\partial\theta_{k,(i)}} \ln \pi(a_{(i)}|\theta, s_t) \\
&= \frac{\partial\mu_{(i)}}{\partial\theta_{k,(i)}} \frac{\partial}{\partial\mu_{(i)}} \ln\left(\frac{1}{P_{in}\sigma_{(i)}\sqrt{2\pi}} \exp\frac{(a_{(i)}-\mu_{(i)})^2}{-2\sigma^2_{(i)}}\right) \\
&= \frac{s_i(1-\mu_{(i)})(1+\mu_{(i)})}{2} \\
&\quad \times \left(\frac{a_{(i)}-\mu_{(i)}}{\sigma^2_{(i)}} + P_{in}\frac{\partial}{\partial\mu_{(i)}}\frac{1}{P_{in}}\right) \quad \cdots\cdots\cdots \ (10)
\end{aligned}$$

ただし $k = 1, 2, \cdots 8$ である．$k = 9$ の場合の適正度は

$$\begin{aligned}
e_\pi(t)_{k,(i)} &= \frac{\partial}{\partial\theta_{k,(i)}} \ln \pi(a_{(i)}|\theta, s_t) \\
&= \frac{\partial\sigma_{(i)}}{\partial\theta_{k,(i)}} \frac{\partial}{\partial\sigma_{(i)}} \ln\left(\frac{1}{P_{in}\sigma_{(i)}\sqrt{2\pi}} \exp\frac{(a_{(i)}-\mu_{(i)})^2}{-2\sigma^2_{(i)}}\right) \\
&= \sigma_{(i)}(1-\sigma_{(i)}) \\
&\quad \times \left(\frac{(a_{(i)}-\mu_{(i)})^2 - \sigma^2_{(i)}}{\sigma^3_{(i)}} + P_{in}\frac{\partial}{\partial\sigma_{(i)}}\frac{1}{P_{in}}\right) , \quad (11)
\end{aligned}$$

ただし $k = 9$ である．式 10 と式 11 の第 2 項は定積分を $\mu_{(i)}$ または $\sigma_{(i)}$ で微分する操作が求められる．本論文ではこれを近似的な数値計算によって与えたが，これらの関数は，上下界を持つ $\mu_{(i)}$ や $\sigma_{(i)}$ にのみ依存するので，テーブル等を用いることで容易に計算コストを減らせる．

ここで $\sigma_{(i)}$ が 0 に近付くと適正度が発散してしまうことに注意が必要である．これは式 10，11 において $\sigma_{(i)}$ が分母になっているためである．適正度が発散すると政策更新のステップ幅が不当に大きくなりすぎて学習が失敗してしまう．そのため本論文では更新のステップサイズを $\sigma^2_{(i)}$ に比例させるというヒューリスティクスを用いた．このとき適正度は以下のように与えられる．

$$\begin{aligned}
e_\pi(t)_{k,(i)} &= \frac{s_i\,(1-\mu_{(i)})(1+\mu_{(i)})}{2}\left(a_{(i)}-\mu_{(i)}\right) \\
&\quad + \frac{s_i\,(1+\mu_{(i)})(1-\mu_{(i)})^2}{\sigma}_{(i)} \left(P_{in}\frac{\partial}{\partial\mu_{(i)}}\frac{1}{P_{in}}\right) , \quad (12) \\
&\quad \text{where } k = 1, 2, \cdots 8, \\
&= (1-\sigma_{(i)})\left((a_{(i)}-\mu_{(i)})^2 - \sigma^2_{(i)}\right) \\
&\quad + (1-\sigma_{(i)})\sigma^3_{(i)} \left(P_{in}\frac{\partial}{\partial\sigma_{(i)}}\frac{1}{P_{in}}\right) , \text{where } k = 9.
\end{aligned}$$

4. 実験結果

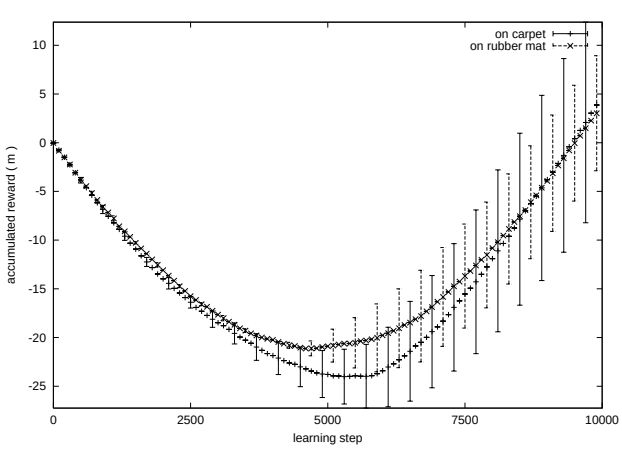

図 4　カーペット上とゴムマット上での学習の様子．各試行は 10000 ステップ (約 80 分)．

Fig. 4. Learning curves averaged over 2 trials on the carpet and the high-frictional rubber mat. Each run consisted of 10000 steps (about 80 minutes).

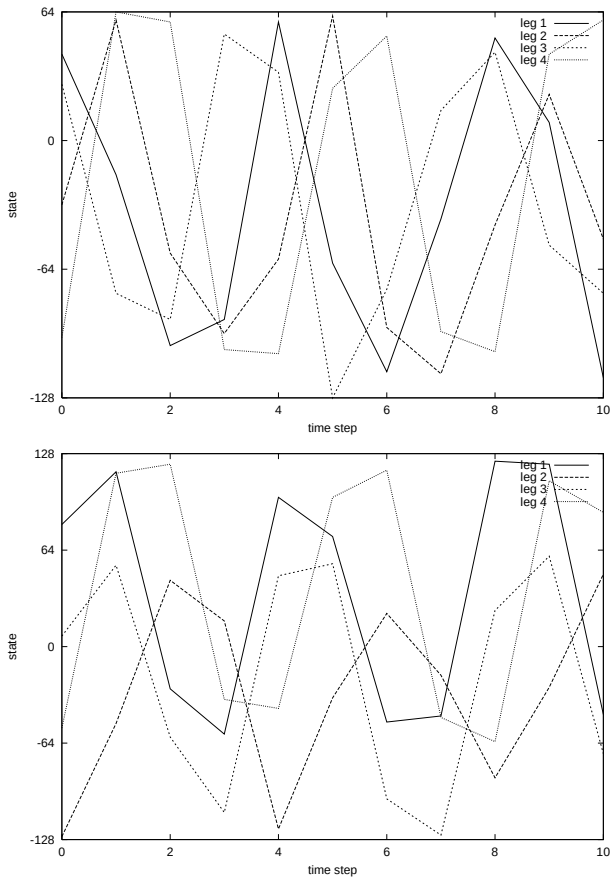

図 5　カーペット上にて 10000 ステップ学習後 (約 80 分) に得た動作の歩容．上はリフト脚，下はスイング脚の動き．

Fig. 5.　Gait pattern of the learned behavior on the carpet after 10000 steps (about 80 minutes). The top graph shows the movement of lifting legs, and the bottom graph is that of swinging legs.

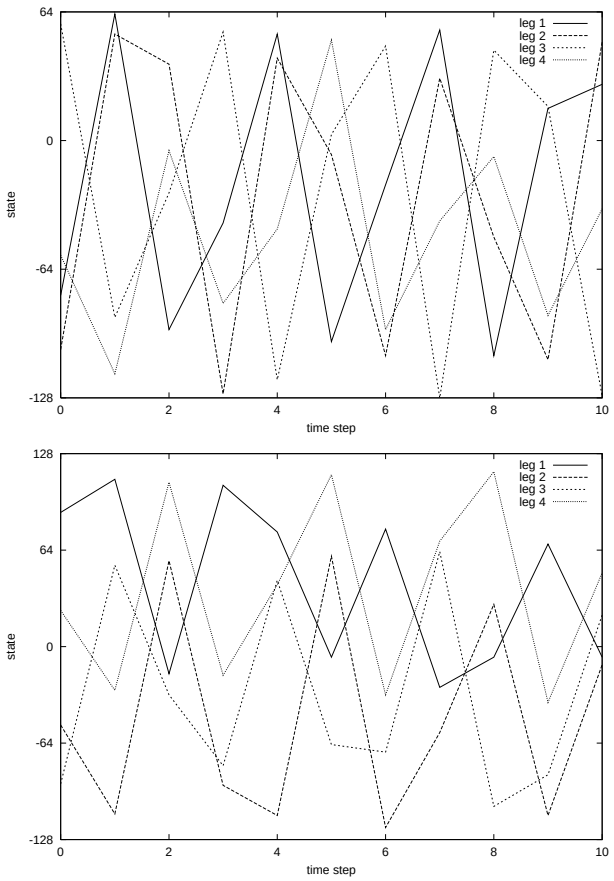

図 6　ゴムマット上にて 10000 ステップ学習後 (約 80 分) に得た動作の歩容．上はリフト脚，下はスイング脚の動き．

Fig. 6.　Gait pattern of the learned behavior on the rubber mat after 10000 steps (about 80 minutes). The top graph shows the movement of lifting legs, and the bottom graph is that of swinging legs.

比較的よく滑べるカーペット上および摩擦の大きなゴムマット上の 2 種類の条件下で実験を行った．パラメータ設定は $\gamma = 0.9$，$\alpha_v = 0.1$，$\alpha_\pi = 0.002$，$\lambda_v = 1.0$，$\lambda_\pi = 1.0$．

Fig.4 はカーペットおよびゴムマット上における学習曲線を表す．縦軸は報酬の累積値を前進した距離に換算した値を表すが，報酬は (1 ステップで 2 つの車輪が前進した平均値) − (1 ステップでの 2 つの車輪の差分の絶対値) という形式で与えられるため，学習初期においてグラフが下降しているのはロボットがバックしているせいではなく，ボディが左右に旋回しているためである．ロボットはカーペットおよびゴムマット上の両方の設定において 6000 ステップ (約 50 分) 以下で良好な挙動を獲得している．10000 ステップ後の実際の移動速度は約 5cm/sec である．グラフ上に現れているカーペットおよびゴムマット上での性能の違いは，単にカーペットとゴムマットの大きさの違いに起因する．ロボットがマットの端まで移動すると，観測者はロボットを持ち上げて向きを変えるが，この期間もロボットは学習を続けている．ロボットが持ち上げられている間は報酬が入らない．そのため，比較的小さいゴムマット上では頻繁に持ち上げる必要があるため，トータルで報酬が少なくなってしまったものである．

どちらの床でも，ロボットは Fig.7, 8 に示すようにカメのような歩行動作を獲得した．Fig.5, 6 はその歩容 (gait) を示す．右側の前足 (leg4) と左側の後ろ足 (leg2) はほぼ同位相で，また左側の前足 (leg1) と右側の後ろ足 (leg3) がほぼ同位相で同期して動いており，歩容はトロット (trot) に近い．リフト脚は足を上に持ち上げる方向が + である．

カーペット上とゴムマット上では獲得された動作にはやや違いが見られた．カーペット上では，ボディは容易に床をすべることが可能なのでしばしばボディを引きずる動作が見られた．これは，支持脚で体重を支える動作よりも前進させる動作を優先して学習した結果と考えられる．一方，摩擦の大きなゴムマット上では，ロボットのボディが床から離れた状態を保持し続ける傾向が見られた．これらの結果は，ロボットがそれぞれ周囲の状況に応じて適応的に動作を獲得できたことを示している．

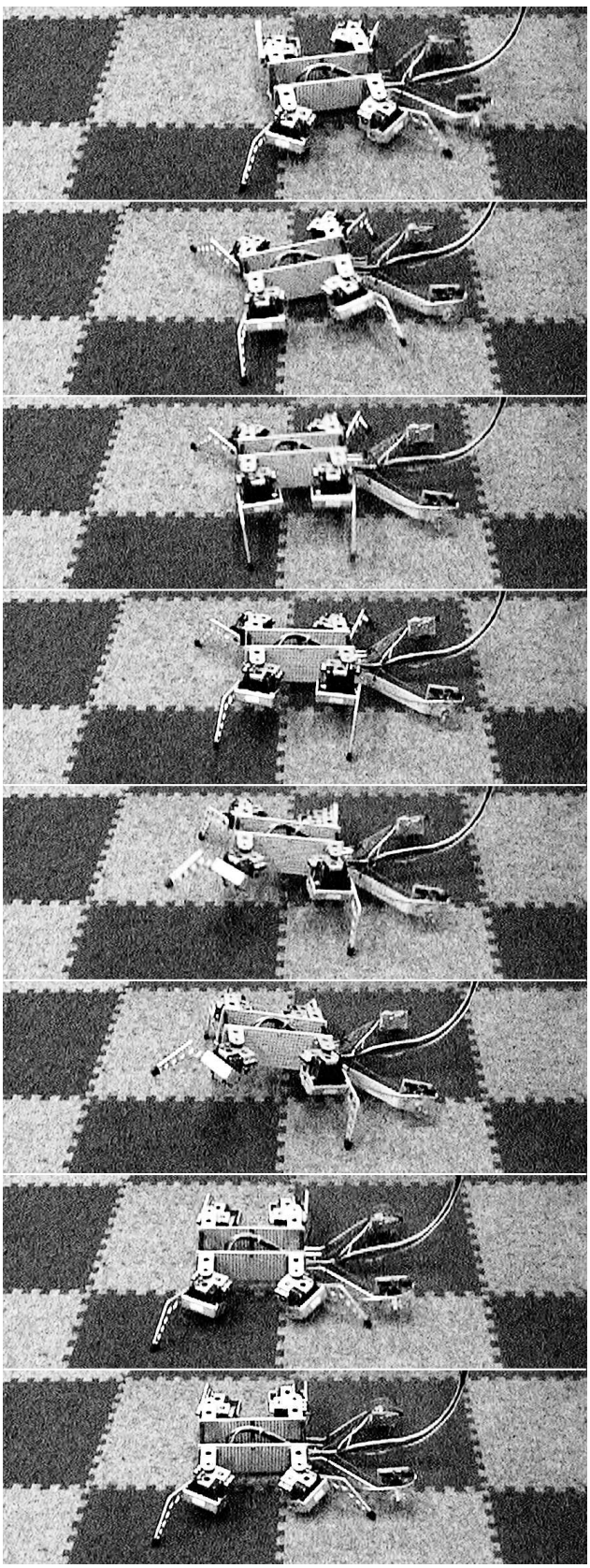

図 7　カーペット上にて 10000 ステップ (約 80 分) 学習後に得た動作の一例．

Fig. 7.　Learned behavior on the carpet after 10000 steps (about 80 minutes).

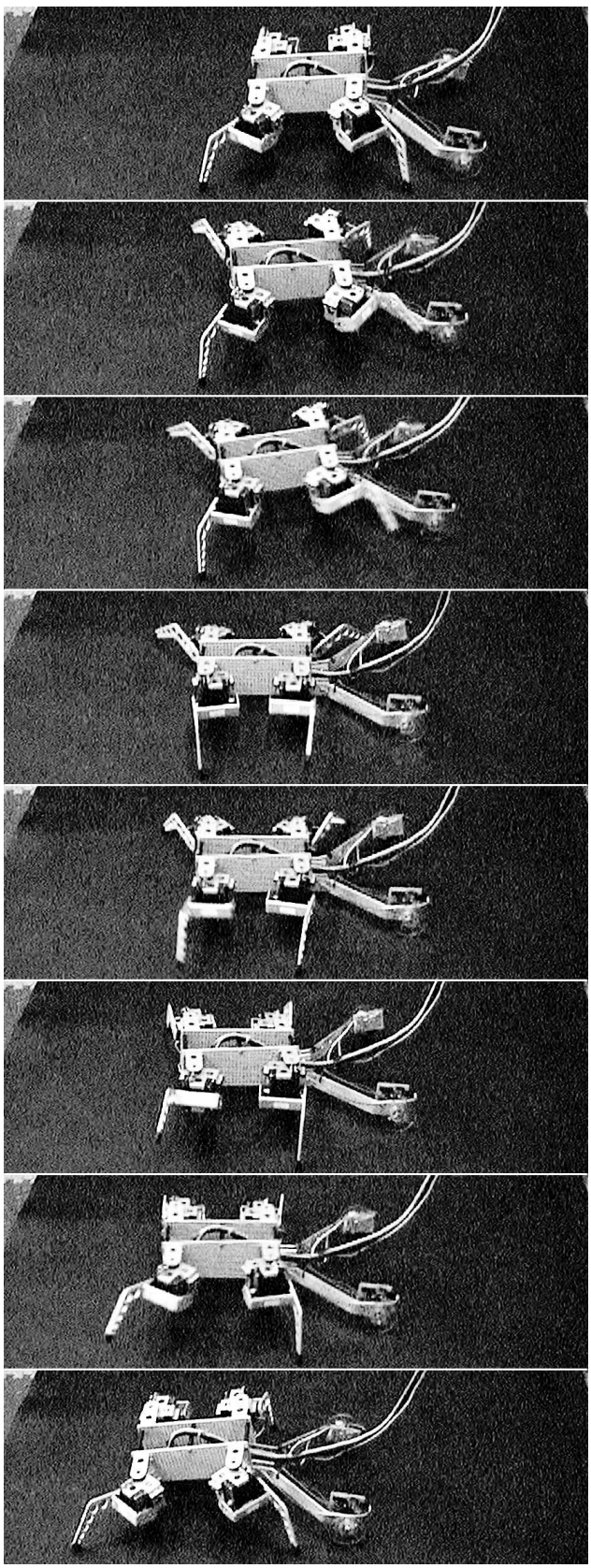

図 8　ゴムマット上にて 10000 ステップ (約 80 分) 学習後に得た動作の一例．

Fig. 8.　Learned behavior on the rubber mat after 10000 steps (about 80 minutes).

5. 考　　察

Actor での行動選択に正規分布を用いる方法は行動選択時に必要な計算と学習時の計算が単純で簡単だという優れた特徴を有している [18] [11] [16] が，本ロボットにそのまま実装すると学習に失敗する．失敗の主な原因は，actor が行動空間の定義域についての情報を利用していないことに起因する．従来の正規分布による行動選択では，actor が定義域外の行動を選んだ場合，ロボットはそれを実行したふりをするが，実際には定義域境界上の行動を実行する．もし分布の平均を示すパラメータが定義域のはるか外側の値になってしまったら，ロボットはランダムな行動をとることがほとんどできなくなり，学習できなくなる．提案手法ではこの問題を単に定義域外の行動選択を棄却することで回避している．しかしその代償として適正度の計算に数値積分の微分演算という余計なコストが要求される．

本実験の critic で用いた粗い分割による量子化は高次元の状態空間を扱う方法として有望だが，非マルコフ性が生じ，またそのように粗く近似された価値関数から良い政策を導くことは困難という問題点がある．本実験の結果は，actor の適正度の履歴が上記の問題点の解決に有効であることを示している．文献 [19] では状態-行動価値関数の近似における状態特徴ベクトルとして actor の適正度を使うべきであると主張している．彼らの手法は理論的な根拠に基づいており，高次元空間に対しても有望なため，この手法との組み合わせることでさらに性能向上が期待できる．

政策勾配定理 (policy gradient theorem) [17] により，actor は政策パラメータで行動選択の分布関数を微分さえできれば政策関数表現に非線形関数でも用いることができる．この政策関数表現における柔軟性は，エキスパートの知識を政策に埋め込む場合に便利である．エキスパートの知識は状態入力とそのとき出力すべき行動のペアで与えられる場合がほとんどなので，教師付き学習のノウハウを応用できる．よって actor に Fuzzy-logic や Neural net などを用いればこのような知識を組み込むことが容易だという実用的特徴がある．それに対し Q-learning など value-based な手法では value 関数近似に linear architecture 以外の方法を用いると更新アルゴリズムが極めて複雑になる．設計者の知識を value 関数という形式で表現することも困難である．

本実験では，政策表現にシグモイド関数や正規分布を利用する部分が「エキスパートの知識の組み込み」に相当する．このおかげで少ないパラメータを用いて妥当な政策表現が可能となり，探索すべき空間が小さくなり，実時間で学習できたと考えられるが，同時に政策表現の柔軟性が制約される．

獲得された動作では，ボディを引きずるなど好ましくないように思える動作が見られた．ボディが床についているかどうかや，エネルギーなどを報酬に反映させればよりロボティクス的に好ましい動作が獲得できるだろう．

(平成 13 年 7 月 24 日受付，同 13 年 10 月 18 日再受付)

文　　献

(1) 木村 元，宮崎 和光，小林 重信：強化学習システムの設計指針，計測と制御, Vol.38, No.10, pp.618–623 (1999).

(2) Watkins, C. J. C. H. & Dayan, P.: Technical Note: *Q*-Learning, *Machine Learning 8*, pp.279–292 (1992).

(3) 堀内 匡，藤野 昭典，片井 修，椹木 哲夫：連続値入出力を扱うファジィ内挿型 Q-learning の提案，計測自動制御学会論文集, Vol.35, No.2, pp.271–279 (1999).

(4) Sutton, R. S. & Barto, A.: Reinforcement Learning: An Introduction, *A Bradford Book*, The MIT Press (1998).

(5) Bertsekas, D.P. & Tsitsiklis, J.N.: *Neuro-Dynamic Programming*, Athena Scientific (1996).

(6) Tsitsiklis, J. N., & Roy, B. V.: An Analysis of Temporal-Difference Learning with Function Approximation, IEEE Transactions on Automatic Control, Vol.42, No.5, pp.674–690 (1997).

(7) Santamaria, J.C., Sutton, R.S. & Ram, A.: Experiments with Reinforcement Learning in Problems with Continuous State and Action Spaces, *Adaptive Behavior 6 (2)*, pp.163–218 (1998).

(8) 深尾 隆則，稲山 典克，足立 紀彦：正則化理論を用いた連続的状態と行動を扱う強化学習，システム制御情報学会論文誌，Vol.11, No.11, pp.593–599 (1998).

(9) Moore A.W. & Atkeson, C.G.: The Parti-game Algorithm for Variable Resolution Reinforcement Learning in Multidimensional State-spaces, *Machine Learning 21*, pp.199–233 (1995).

(10) 矢入 健久，堀 浩一，中須賀 真一: 複数行動結果を考慮した最尤推定に基づく状態一般化法，人工知能学会誌, Vol.16, No.1, pp.128–138 (2001).

(11) Doya, K.: Efficient Nonlinear Control with Actor-Tutor Architecture, *Advances in Neural Information Processing Systems 9*, pp. 1012–1018 (1997).

(12) Morimoto, J. & Doya, K.: Acquisition of Stand-up Behavior by a Real Robot using Hierarchical Reinforcement Learning, *Proceedings of the 17th International Conference on Machine Learning*, pp.623–630 (2000).

(13) Barto, A.G., Sutton, R.S. & Anderson, C.W.: Neuronlike adaptive elements that can solve difficult learning control problems, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC–13, no.5, pp. 834–846 (1983).

(14) Singh, S. P. & Sutton, R.S.: Reinforcement Learning with Replacing Eligibility Traces, *Machine Learning 22*, pp.123–158 (1996).

(15) Baird, L. & Moore, A.: Gradient Descent for General Reinforcement Learning, *Advances in Neural Information Processing Systems 11*, pp.968–974 (1999).

(16) 木村 元，小林 重信：Actor に適正度の履歴を用いた Actor-Critic アルゴリズム– 不完全な Value-Function のもとでの強化学習，人工知能学会誌, Vol.15, No.2, pp.267–275 (2000).

(17) Sutton, R. S., McAllester, D., Singh, S. & Mansour, Y.: Policy Gradient Methods for Reinforcement Learning with Function Approximation, *Advances in Neural Information Processing Systems 12 (NIPS12)*, pp. 1057–1063 (2000).

(18) Williams, R. J.: Simple Statistical Gradient Following Algorithms for Connectionist Reinforcement Learning, *Machine Learning 8*, pp. 229–256 (1992).

(19) Konda, V.R. & Tsitsiklis, J.N.: Actor-Critic Algorithms, *Advances in Neural Information Processing Systems 12*, pp. 1008–1014 (2000).

付　　録

木　村　　元（正員）1997 年東京工業大学大学院知能科学専攻博士課程修了，同年 4 月日本学術振興会 PD 研究員，1998 年 4 月，東京工業大学大学院総合理工学研究科助手，現在に至る．人工知能，特に強化学習に関する研究に従事．

山　下　　透（非会員）2001 年東京工業大学大学院知能システム科学専攻修士課程修了，同年 4 月任天堂（株）に所属．

小　林　重　信（非会員）1974 年東京工業大学大学院博士課程経営工学専攻修了．同年 4 月，同大学工学部制御工学科助手．1981 年 8 月，同大学大学院総合理工学研究科助教授．1990 年 8 月，教授．現在に至る．問題解決と推論制御，知識獲得と学習などの研究に従事．