強化学習における基底関数の適応的配置の一手法*

飯田 信吾† 加納 政芳† 加藤 昇平† 伊藤 英則†
† 名古屋工業大学

1 はじめに

近年，ヒューマノイドロボット (HR) の制御手法として強化学習が注目されている．従来，強化学習で扱ってきた問題は離散状態空間である．しかし，HR を制御するためには連続状態空間を扱う必要がある．強化学習において連続状態を扱うための手法として，基底関数ネットワークがあげられる．このネットワークは次元数が少ない場合には有効であるが，HR の制御のような多次元問題においては，基底関数の数が膨大となる．これを解消するための手法として，必要に応じて基底関数を配置する手法が提案されている [1]．本稿では同手法に対し，必要に応じて配置した基底関数を削除することで局所解への収束を抑制する手法を提案する．

2 Actor-Critic 法

本研究では，HR 制御のための強化学習法として，Actor-Critic 法を使用する．Actor-Critic 法は，Actor と呼ばれる制御器と Critic と呼ばれる評価器で構成される．同手法の構造を，図 1 に示す．本研究では，Actor および Critic の学習に対して基底関数ネットワークを用いる．

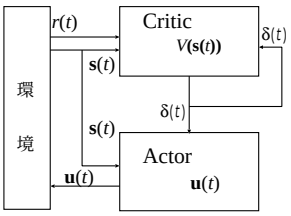

図 1: Actor-Critic 法の構造

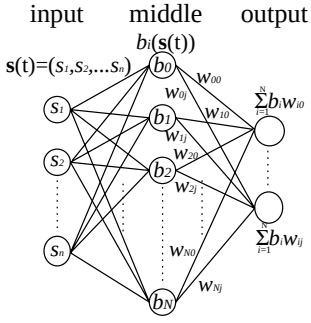

図 2: 基底関数ネットワーク

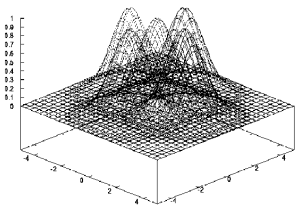

図 3: 動径基底関数

基底関数ネットワークは図 2 のように 3 層構造を有しており，基底関数 $b_i(\boldsymbol{s}(t))$ を中間層に持つ．本研究では，動径基底関数 (図 3) を正規化した正規化ガウス関数を基底関数とするネットワークを用いる．同関数は，動径基底関数よりも汎化能力に優れており，少ない基底関数で広範囲をカバーできるため高次元状態空間に適している．

正規化ガウス関数 $b_i(\boldsymbol{s}(t))$ は以下の式で定義される．

$$b_i(\boldsymbol{s}(t)) = \frac{a_i(\boldsymbol{s}(t))}{\sum_{k=1}^{N} a_k(\boldsymbol{s}(t))}. \quad (1)$$

ただし，N は基底関数の数である．$a_i(\boldsymbol{s}(t))$ は動径基底関数であり，次式で表される．

$$a_i(\boldsymbol{s}(t)) = \exp(-\frac{1}{2}\|M(\boldsymbol{s}(t) - \boldsymbol{c}_i)\|). \quad (2)$$

ただし，$\boldsymbol{c}_i$ は基底関数の中心，M は基底関数の形状を決定する行列である．

基底関数ネットワークを用いた Actor-Critic 法は，図 4 の処理を繰り返すことで，Critic は価値関数 $V(\boldsymbol{s})$ を正しく推定し，Actor は $V(\boldsymbol{s})$ を最大化する行動を選択するように学習する．

i) 状態 $\boldsymbol{s}(t)$ を環境から観測する．Actor は，行動 $\boldsymbol{u}(t)$ における j 番目の行動出力 $u_j(t)$ を以下の式により算出する．
$$u_j(t) = u_j^{\max} g\left(\textstyle\sum_i \omega_{ij} b_i(\boldsymbol{s}(t)) + n_j(t)\right).$$
ただし，$u_j^{\max}$ は j 番目の行動出力の最大値，$b_i()$ は基底関数，ω_{ij} は荷重，$n_j(t)$ はノイズ関数である．また，$g()$ は制御出力を最大制御出力に収束させるためのシグモイド関数である．

ii) Critic は報酬 $r(t)$ を受け取り，次の状態 $\boldsymbol{s}(t+\Delta t)$ を観測する．Critic は，TD 誤差 $\delta(t)$ を以下の式により算出する．
$$\delta(t) = r(t) + \gamma V(\boldsymbol{s}(t+\Delta t)) - V(\boldsymbol{s}(t)).$$
ただし，γ は割引率，$V(\boldsymbol{s})$ は推定価値関数であり以下の式により算出される．
$$V(\boldsymbol{s}(t)) = \textstyle\sum_i^N v_i b_i(\boldsymbol{s}(t)).$$
v_i は荷重である．

iii) Actor は，TD 誤差に基づいて荷重 ω_{ij} を更新する．
$$\omega_{ij} \leftarrow \omega_{ij} + \beta\delta(t) n_j(t) b_i(\boldsymbol{s}(t)).$$
ただし，β は学習率である．

iv) Critic は荷重 v_i を更新する．
$$v_i \leftarrow v_i + \alpha\delta(t) e_i,$$
$$e_i \leftarrow \gamma\lambda e_i + b_i(\boldsymbol{s}(t)).$$
ただし，α は学習率，e_i は適格度トレース，λ はトレース減衰パラメータである．

v) 時刻を更新する．
$$t \leftarrow t + \Delta t.$$

図 4: 基底関数ネットワークを用いた ActorCritic 法

3 適応的配置手法

本稿では，基底関数の適応的配置手法を提案する．本手法は，2 節の Actor-Critic 法に，動径基底関数 $a_i(\boldsymbol{s})$ の活性度の履歴 ε_i[2]，追加抑制時間 η，および基底関数の存在時間 τ_i を導入したものである．ε_i および τ_i は基底関数の数だけ準備し，η は Actor，Critic のネットワークごとに 1 つずつ準備する．

*A Method for Dynamic Allocation of Basis Functions in Reinforcement Learning,
Shingo IIDA†, Masayoshi KANOH†, Shohei KATO† and Hidenori ITOH†.
† Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya 466-8555, Japan.

追加抑制時間 η は，削除処理を行った後，すぐさま，削除された基底関数とほぼ同じ状態に基底関数を追加することを抑制するものである．基底関数の追加条件として文献 [1] の条件に，η による条件を加えたものを定義する．

定義 3.1 追加条件 Actor，Critic のネットワークにおいて以下の条件を満たすとき，基底関数を $\boldsymbol{c} = \boldsymbol{s}(t)$ の位置に追加配置する．

$$\begin{aligned} \max_i a_i(\boldsymbol{s}(t)) < a_{\min} \quad &\text{and} \quad |h(t)| > \delta_{\max} \\ &\text{and} \quad \eta > T_{\text{add}} \end{aligned} \tag{3}$$

ただし，$a_{\min}, \delta_{\max}, T_{\text{add}}$ はしきい値，$h(t)$ は Actor では $h(t) = \delta(t) n_j(t)$，Critic では $h(t) = \delta(t)$ である．□

動径基底関数の活性度の履歴 ε_i の更新は，以下の式で行われる．

$$\varepsilon_i \leftarrow \kappa \varepsilon_i + a_i(\boldsymbol{s}(t)). \tag{4}$$

ただし，κ は減衰率である．ε_i は，HR が最近取った状態を評価することができる．仮に，HR の状態評価・行動が局所解に陥っているならば，その状態付近に配置されている基底関数の活性度は高くなるため，ε_i の値は大きなものとなる．これより，基底関数の削除条件を以下のように定義する．

定義 3.2 削除条件 Actor，Critic のネットワークにおいて以下の条件を満たすとき，基底関数 $b_i(\boldsymbol{s}(t))$ を削除する．

$$\varepsilon_i > \varepsilon_{\max} \quad \text{and} \quad \tau_i > T_{\text{erase}} \tag{5}$$

ただし，$\varepsilon_{\max}, T_{\text{erase}}$ はしきい値である．□

存在時間 τ_i は，追加した基底関数がすぐに削除されることを抑制するものである．また，基底関数を削除した際に追加抑制時間 η は 0 に初期化される．

4 椅子からの立ち上がり動作学習実験

本手法の有効性を確認するために，HR の椅子からの立ち上がり動作の学習を行う（図 5）．本実験では，学習シミュレーションのために，富士通の HOAP-1（図 6）の機構データからモデルを作成した．学習シミュレーションには，Open Dynamics Engine[3] を用いた．

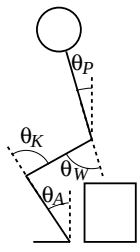

図 5: 学習動作

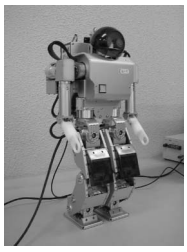
図 6: HOAP-1

Actor の行動出力は，腰，膝および足首の角度変化量 $(\Delta\theta_W, \Delta\theta_K, \Delta\theta_A)$ とした．基底関数ネットワークへの入力は，$(\theta_W, \dot{\theta}_W, \theta_K, \dot{\theta}_K, \theta_A, \dot{\theta}_A, \theta_P, \dot{\theta}_P)$ とした（図 5 参照）．学習パラメータは，$\alpha = 0.02$, $\beta = 0.1$, $\gamma = 0.9$, $\lambda = 0.6$, $\kappa = 0.9$, $\varepsilon_{\max} = 5.0$, $\delta_{\max} = 0.5$, $a_{\min} = 0.4$, $T_{\text{add}} = 1[s]$, $T_{\text{erase}} = 3[s]$, $\Delta t = 0.01[s]$, $u_j^{\max} = \frac{1}{36}\pi[rad]$, $M = \text{diag}(2.0, 0.57, 2.0, 0.57, 2.0, 0.57, 2.0, 0.57)$ とした．報酬 $r(t)$ は以下の式で与えた．

$$r(t) = -\left|\frac{y - l_s}{l_s - l_d}\right| \tag{6}$$

ただし，y は胸の重心位置，l_s は起立時の胸の重心位置，l_d は転倒時の胸の重心位置である．$l_s = 0.351, l_d = 0.20$ とした．1 試行は，実行時間が 10 秒経過するか，転倒したら終了とした．

5 評価

上記の条件下で実験を行ったところ，2132 回目の試行で図 7 に示す動作を獲得することができた．このときの基底関数の数は Actor，Critic ともに 72 個であった．

つぎに，基底関数削除の効果を検証するために，手法 [1] との比較実験を行った．図 8 に Actor における基底関数の数の変化を示す．同図より，本手法の方が少ない基底関数で学習に成功していることがわかる．

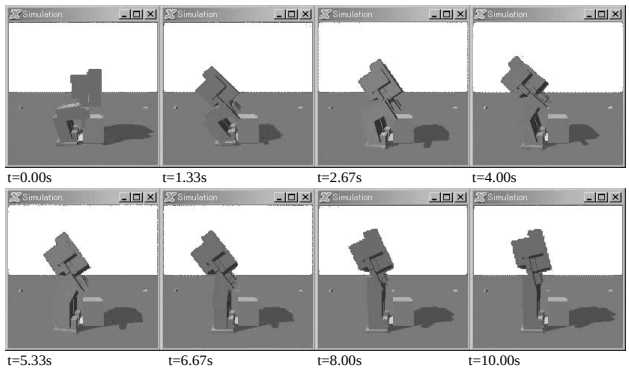

図 7: 実験結果

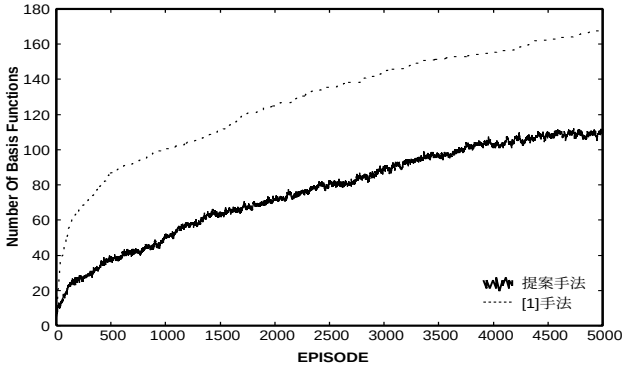

図 8: Actor における基底関数の数の変化 (5 回平均)

6 おわりに

本稿では，強化学習の基底関数を適応的配置する手法を提案した．本手法の有効性を確認するために，HR に椅子からの立ち上がり動作を学習させたところ，同動作の学習に成功した．実験により，基底関数の削減に対する有効性は確認できたが，局所解への収束に関する考察を行う必要がある．

参考文献

[1] 森本 淳，銅谷 賢治：強化学習を用いた高次元連続空間における系列運動学習-起き上がり運動の獲得-，電子情報通信学会論文誌, Vol.J82-D-II,No.11,pp.2118-2131(1999)

[2] 近藤 敏之，伊藤 宏司：オンライン進化的強化学習アルゴリズムの提案，計測自動制御学会，第 13 回自律分散システムシンポジウム，p.85-90 (2001)

[3] Russell Smith：Open Dynamics Engine, http://opende.sourceforge. net/ode. html.