解 説

ロボットの強化学習における状態-行動空間の汎化

Generalization of state-action space in reinforcement learning robots

木村 元* 小林重信 *東京工業大学 大学院総合理工学研究科
Hajime Kimura* *and* Shigenobu Kobayashi *Interdisciplinary Graduate School of Sci. and Eng., Tokyo Institute of Technology

1. はじめに

強化学習は，報酬という単純な指示から，それよりずっと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 [5]．強化学習で最も一般的なのは Q-learning [13] のように離散化された状態-行動対の評価値 (value) を推定した後に最適な政策を得る方法である．強化学習を実ロボットへ適用するとき，状態・行動表現の与え方は設計上大きな問題だが [1] ，本稿では連続な空間の扱いに焦点をあてる．連続な状態空間を扱う強化学習では，Q-learning と汎化テクニックと組み合わせる方法 [11] が知られている．しかし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だけでなく，連続な行動空間の扱いも求められるため，Q-learning によって全ての状態-行動空間における評価値を推定することは，メモリ容量の点からも探索のための試行回数の点でも事実上実行不能である．

一方，actor-critic アルゴリズム [11] は，政策勾配法に基づく強化学習法で，連続な状態-行動空間への適用が比較的容易である．この手法は，ある行動政策（状態から行動への分布関数）のもとでの各状態の評価値を推定し，これを手がかりにしてその行動政策を改善していくことを基本としている．Q-learning のように全状態-行動の評価値を推定してから最適な状態-行動対を見出すわけではないため，少ないメモリーでの実装が可能であり，学習も高速である．さらに，行動選択を特徴付ける政策関数には任意の確率分布関数または確率密度関数が使用できるため，ガウス分布など計算が簡単な分布を用いれば連続な行動空間への適用が容易である．このアプローチは局所解に陥る可能性を伴うものの，仕組みが単純な割に強力なため，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 [9] [10]．図 6 は著者らが取り組んだ 4 足ロボットへの actor-critic の適用例 [7] で，現在の関節の角度を状態入力として，次のとるべき関節角度の目標値を行動として出力する．各時間ステップにおいて，まっすぐ高速に前進すると最大の報酬が与えられるよう設定しておくと，強化学習エージェントは試行錯誤を繰返すうちに図のような足の運び方を見出す．

本稿では actor-critic アルゴリズムを概説し，連続な状態-行動空間における強化学習問題への適用について簡単な例を示す。強化学習における評価値の定義や最適性，環境の数理モデル，Q-learning など一般的解説については，文献 [11] [5] [2] 等を参照されたい．

2. 強化学習の枠組み

2.1 Actor-Critic アルゴリズム

図 1 に本手法の概要を示す．

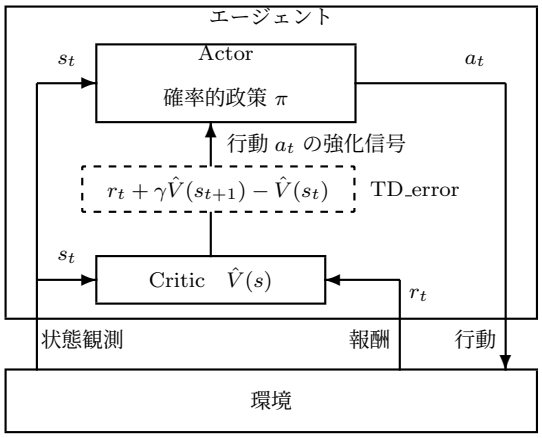
【行動選択】において確率的政策を特徴付ける関数 $\pi(a|s)$ は，状態 s において行動 a を選択する確率（または確率密度）をあらわす．Actor は状態 s_t に応じた確率的政策 π に従って行動 a_t を実行する．

【実行した行動の評価】において，critic は actor の政策のもとでの各状態 s の評価値 $V(s)$ を推定する．評価値 $V(s)$ は，時刻 $t=0$ のとき状態 s からスタートした場合の割引報酬合計の期待値 $E\{r_0+\gamma r_1+\cdots+\gamma^t r_t+\cdots|\pi,s\}$ である．ただし γ は割引率を表し，$0\leq\gamma<1$ である．Actor への強化信号として計算される TD-error は，状態遷移による状態評価値の変化を示す．

【行動の強化】において actor は，critic で計算される TD_error を行動評価値として行動選択確率を更新することにより，政策関数 $\pi(a|s)$ を改善する．TD_error が正のとき，エージェントは良い状態へ遷移したと考えられるので，状態 s_t における行動 a_t の選択確率を増やす．逆に TD_error が負のとき，状態 s_t における行動 a_t の選択確率を減らす．この原理は行動空間が連続であってもそのまま拡張できる．行動選択確率を特徴付ける確率的政策関数 $\pi(a|s)$ は，政策パラメータベクトル θ を用いて表され，θ を調節することで行動選択確率を変化させる．より洗練されたアルゴリズムでは，以下のように政策パラメータ θ を更新する [6] [12]：

$$\theta \leftarrow \theta+\alpha_\pi \mathrm{TD\_error}\frac{1}{\pi(a_t|s_t)}\frac{\partial\pi(a_t|s_t)}{\partial\theta} \qquad (1)$$

原稿受付
キーワード：reinforcement learning, continuous space, generalization, actor-critic
*横浜市緑区長津田 4259
*4259 Nagatsuta Midori-ku Yokohama, Japan



(1) 【行動選択】Actor は状態 s_t に応じた確率的政策 π に従って行動 a_t を実行する．
(2) 【実行した行動の評価】Critic は報酬 r_t を受取り，次状態 s_{t+1} を観測し，以下の TD_error を計算

$$(\text{TD\_error}) = \left[r_t + \gamma\,\hat{V}(s_{t+1})\right] - \hat{V}(s_t)\ ,$$

γ は割引率，$V(s)$ は critic が推定した割引報酬の期待値．
(3) 【行動の強化】TD_error を用いて行動選択確率を更新
(TD_error) > 0 ならば，実行した行動 a_t は比較的好ましいものと考えられるので，この選択確率を増やす．
(TD_error) < 0 ならば，実行した行動 a_t は好ましくないと考えられるので，この選択確率を減らす．
(4) 【状態評価値の更新】TD 法を用いて critic の value の推定値を更新．例えば TD(0) ならば以下のように計算：
$\hat{V}(s_t) \leftarrow \hat{V}(s_t) + \alpha\,(\text{TD\_error})$, ただし α は学習率．
(5) 手順 (1) から繰り返す．

図 1 一般的な actor-critic アルゴリズム

ただし α_π は学習係数を表す．このとき

$$e_t = \frac{1}{\pi(a_t|s_t)}\frac{\partial\pi(a_t|s_t)}{\partial\theta} = \frac{\partial \ln \pi(a_t|s_t)}{\partial\theta}$$

上記の e_t は適正度 (eligibility) [14] と呼ばれる．

近年では，この部分に自然勾配 (natural gradient) を用いて更新する方法 [10] が提案されている．

【状態評価値の更新】critic は actor の政策のもとでの各状態 s の評価値 $V(s)$ を推定する．TD_error を用いて更新するのが一般的である．

2.2 観測入力からの状態表現生成

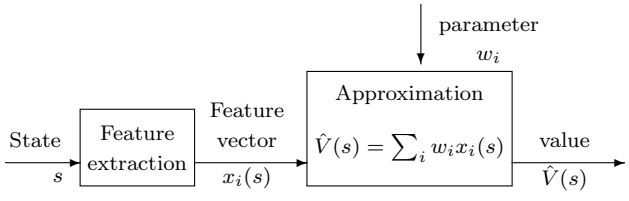


図 2 特徴量ベクトルへの変換を介した線形アーキテクチャによる汎化．

強化学習の基本は，センサ入力（観測入力）より生成された状態表現から，状態評価値や行動確率分布への写像関数を表現することである．この観測入力からの状態表現生成は特徴量抽出 (feature extraction) に関連するが，強化学習ではこの部分については扱わず，予め与えられるものとするのが一般的である．観測入力が連続空間の場合，このテクニックは汎化 (generalization) や関数近似と呼ばれ，強化学習に限らずニューラルネットワークなど幅広い分野で研究されており，強化学習または DP と関数近似法を組み合わせた計算法は neuro-dynamic programming [3] と呼ばれる．Q-learning の最適解への収束が保障される簡便な汎化方法として線形アーキテクチャ(linear architecture) がある．これは固定された基底関数の集合を用いて関数近似を行うもので，ラジアル基底やシグモイド関数を用いたものなど様々だが，状態空間中の座標を基底関数によって特徴量ベクトルに変換し，このベクトルの線形重み付け和で関数を表現する点は同じである（図 2）．特徴量ベクトルの生成方法は問題に応じて設計者が与える必要があり，ベクトルのノルムが常に 1 となることが望ましい．図 3 は最も単純な特徴ベクトル生成方法の例を示す。２次元空間を１枚のタイル（グリッド）によって分割し，各タイル要素に特徴量ベクトルの要素を割り当てる．状態入力があるタ

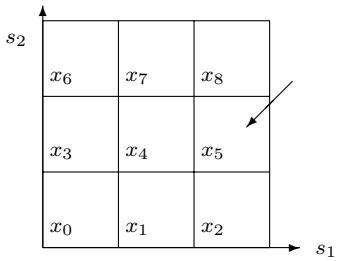


図 3 タイルコーディングによる特徴量ベクトル生成例．状態入力 (s_1, s_2) が矢印の位置のとき，特徴量ベクトル $(x_0, x_1, \cdots x_8)$ は，該当するタイル要素 $x_5 = 1$，それ以外の要素は全てゼロ．

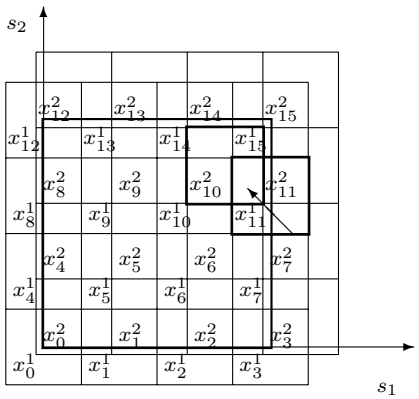


図 4 ２枚のタイルをずらせて重ねる CMAC による特徴量ベクトル生成．大きな太線の枠は状態の定義域，２つの小さい太線枠は状態入力に対応するタイルを表す．状態入力が矢印の位置のとき，特徴量ベクトルは，該当するタイル要素 $x^1_{11} = 0.5$，$x^2_{10} = 0.5$，それ以外の全要素はゼロ．

イル要素内にあるとき，該当する特徴量ベクトル要素の値は 1 に，それ以外の要素はゼロとする．V 値を表現するときは，特徴ベクトル要素と同数のパラメータ w_i^a を用いて線形関数表現する：

$$\hat{V}(s) = \sum_{i=1}^{n} w_i x_i(s) \qquad (2)$$

ただし n は特徴ベクトル要素の個数．線形アーキテクチャによる V 値更新は，単純な勾配法に基づきパラメータ w_i を更新することで達成される：

$$w_i \ \leftarrow \ w_i + \alpha \frac{\partial \hat{V}(s)}{\partial w_i} \mathrm{TD\_error} \qquad (3)$$

式 (2) より $\partial \hat{V}(s)/\partial w_i = x_i$ なので，

$$w_i \ \leftarrow \ w_i + \alpha \ x_i \ \mathrm{TD\_error} \qquad (4)$$

つまり線形アーキテクチャを用いた更新では，更新するパラメータ w_i に対応する特徴の要素の値 x_i だけを使って簡単に更新できる．図 3 のような単純なグリッド分割による線形アーキテクチャの更新式は離散的 TD 法の更新式と等価になる．図 4 はグリッド分割を拡張した CMAC による特徴量生成の例を示す．

ここで紹介した状態表現は全て設計者が予め与える必要があるため，状態空間が高次元になると次元の呪い問題と呼ばれる状態爆発が生じる．そこで，経験した状態のみを状態表現として生成する事例ベース法や，遷移先に応じて状態空間を分割する方法 [15] 等が提案されている．

2.3 連続な行動空間を扱う Actor-Critic

第 2.1 章で述べたように，actor の確率的政策 π を確率分布関数とすることで actor-critic を連続行動空間へ拡張できる．Actor では，一般に確率的政策をパラメータ関数表現し，そのパラメータを勾配法によって更新していく．政策関数の場合，価値関数とは異なりこの部分は必ずしも線形関数である必要はなく，非線形関数を用いることができる．

図 5 は確率的政策 π として以下の式で表される正規分布を用いた例を示す [14].

$$\pi(a|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(a-\mu)^2}{2\sigma^2}\right) \qquad (5)$$

ただし μ は中心値，σ は標準偏差を表す．この正規分布を actor として用いるエージェントは，状態 s に応じて $\mu = \mathrm{mu}(s)$, $\sigma = \mathrm{sigma}(s)$ の正規分布に従ったランダムサンプリングによって行動 a を選択する．状態遷移後，TD_error を計算し，この値に基づいて actor の政策パラメータを更新する．式 5 の正規分布関数の中心値パラメータ μ および標準偏差 σ について式 1 の更新式を適用すると以下の式を得る：

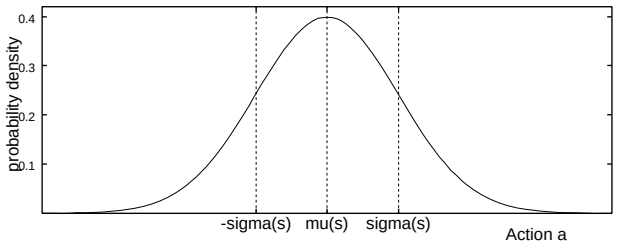


図 5 Actor において 1 次元連続値行動を扱う例．中心値 $\mathrm{mu}(s)$，標準偏差 $\mathrm{sigma}(s)$ の正規分布に従ったランダムサンプルによって行動 a を選択．状態遷移の結果，TD_error が正なら，中心値 $\mathrm{mu}(s)$ を a の方向へ修正．行動 a が $\pm\mathrm{sigma}(s)$ の内側だったなら，$\mathrm{sigma}(s)$ を小さくする方向へ，外側なら大きくする方向へ修正する．TD_error が負なら逆の操作を行う．

$$\frac{1}{\pi(a|\mu,\sigma)} \frac{\partial}{\partial \mu} \pi(a|\mu,\sigma) \ = \ \frac{a-\mu}{\sigma^2} \qquad (6)$$

$$\frac{1}{\pi(a|\mu,\sigma)} \frac{\partial}{\partial \sigma} \pi(a|\mu,\sigma) \ = \ \frac{(a-\mu)^2 - \sigma^2}{\sigma^3} \qquad (7)$$

この式に従った政策パラメータ更新は，直観的には以下のイメージである：TD_error が正のとき，エージェントは良い状態へ遷移したと考えられるので，状態 s_t における行動 a_t の選択確率を高める方向へ修正するが，正規分布の場合は中心値 $\mathrm{mu}(s)$ を a の方向へ修正し，行動 a が $\pm\mathrm{sigma}(s)$ の内側だったなら，$\mathrm{sigma}(s)$ を小さくする方向へ，外側なら大きくする方向へ修正する．TD_error が負の場合は逆の操作を行う．しかし式 6, 7 の更新式をそのまま用いると分母の σ がゼロに近づいたとき値が発散するので，更新のステップ幅を σ^2 に比例させるなどの工夫が必要である [7].

ここで示した連続な行動空間に対する実装例では，1 つの正規分布関数を用いた最も単純な場合を説明した．しかしロボットの問題では，行動空間に上下界が存在する場合が多く，分布に (理論的) 上下界のない正規分布による行動選択が不適当な場合もある．そこで著者らは，行動空間の上下界をはみ出して行動選択しようとした場合には，その行動選択を破棄し，行動空間の上下界に収まる行動を選ぶまで行動選択を繰返す方法を提案し，8 自由度の 4 脚ロボットにおける移動動作獲得に適用し有効性を示した [7]．図 6 はその動作例を示す．この学習器は全関節の角度（8 次元連続値ベクトル）を状態として観測し，約 0.4 秒後の関節角を行動として出力する．行動は各次元毎に上下界のある 1 次元連続行動として扱われ，critic は状態ベクトルの各次元を 2 分割した 256 コのグリッドの特徴量を用いている．

また，ある状態においてとりうる行動が，いくつかの有望な領域に分かれ，それぞれの領域毎に行動選択の調節が必要とされる場合がある．このように複雑な政策表現としては，正規分布を複数用いて重ね合わせる混合正規分布を用いる方法 [16] がある．

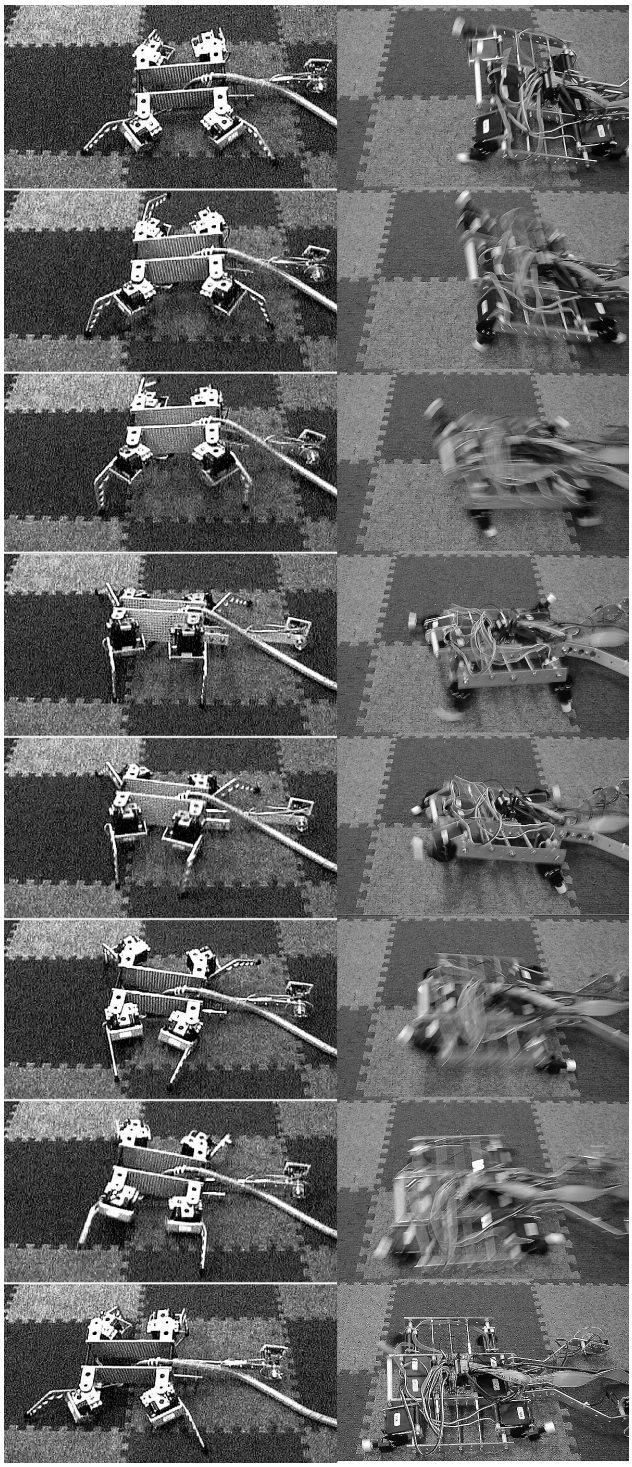


図 6 2 種類の 8 自由度ロボットを強化学習させて得た動作の一例．左側：約 80 分後，右側：約 120 分後．

3. おわりに

本稿では連続な状態-行動空間における強化学習問題に対して最も単純で強力な接近法である actor-critic アルゴリズムを概説し，状態空間汎化の方法と正規分布を用いた連続値の行動選択について紹介した．

参考文献

[1] 浅田 稔：”強化学習の実ロボットへの応用とその課題”，人工知能学会誌, Vol.12, No.6, pp.831–836 (1997).

[2] 阿部 健一：”強化学習–価値関数推定と政策探索”，計測と制御, Vol.41, No.9, pp.680–685 (2002).

[3] Bertsekas, D.P. & Tsitsiklis, J.N.: Neuro-Dynamic Programming, Athena Scientific (1996).

[4] 石井 信，佐藤 雅昭：”統計的手法にもとづく強化学習と制御ルールの獲得”，計測と制御，Vol.39, No.12, pp.763–768 (2000).

[5] 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).

[6] 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).

[7] 木村 元，山下 透，小林 重信：”強化学習による 4 足ロボットの歩行動作獲得”，電気学会 電子情報システム部門誌, Vol.122-C, No.3, pp.330–337 (2002).

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

[9] 森本 淳，銅谷 賢治：”階層型強化学習を用いた 3 リンク 2 関節ロボットによる起立運動の獲得”，日本ロボット学会誌 Vol.19, No.5, pp.574–579, 2001.

[10] Peters, J., Vijayakumar, S. & Schaal, S.: ”Reinforcement Learning for Humanoid Robots - policy gradients and beyond”, 3rd IEEE International Conference on Humanoid Robotics (2003).

[11] Sutton, R.S. & Barto, A.: ”Reinforcement Learning: An Introduction”, A Bradford Book, The MIT Press (1998).

[12] 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).

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

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

[15] 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).

[16] Yoshimoto,J., Ishii,S. & Sato,M.: ”On-line EM reinforcement learning”, In The IEEE-INNS-ENNS International Joint Conference on Neural Networks, III, pp.163–168 (2000).

木村 元

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．

（日本ロボット学会正会員）

小林 重信

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．