NAIST-IS-DT0161028

博士論文

パターン生成器を用いた制御に対する強化学習法

中村 泰

2004年03月24日

奈良先端科学技術大学院大学
情報科学研究科 情報システム学専攻

本論文は奈良先端科学技術大学院大学情報科学研究科に
博士 (工学) 授与の要件として提出した博士論文である。

中村 泰

審査委員：　石井 信 教授
小笠原 司 教授
杉本 謙二 教授
柴田 智広 助教授

パターン生成器を用いた制御に対する強化学習法*

中村 泰

内容梗概

実空間で行動するロボットに対する制御を行う場合, 環境の知識が前もって得られなかったり, 地面などの環境が動的に変化するために, あらかじめ与えられた制御を行うのではなく自律的に環境に適応し, 制御則を獲得する枠組みが必要となる. 強化学習法はこの要求を満たす手法で, 環境や制御対象のシステムなどの先見的な知識が無い場合にも適用できる. しかしながら, 歩行などの複雑な運動に対する制御則の学習は, システムの自由度が大きく非線形性が非常に強いことや, 制御則自体が非線形で複雑となることから困難なものとなる.

一方, 歩行などの生物の運動は, 周期的な信号を生成する中枢パターン生成器 (CPG) とよばれる神経回路によって制御されていることが示唆されている. このような生物の制御機構を参考にして, 周期的な運動に対する CPG を用いた制御法の研究が行われてきた. これらの手法では, 制御器となる CPG を再帰結合型ニューラルネットワークで実装し, 制御対象のシステムからのフィードバック結合やニューロン間の相互結合の重みを変化させることによって制御信号を調整する.

本論文では, CPG を用いた制御に対する自律的な学習の枠組みとして, CPG-actor-critic モデルと呼ばれる新しい強化学習法を提案する. 提案手法は, 複雑な制御則を学習する代りに CPG の結合重みの学習を行う手法であり, 線形制御器に対する学習を行うことで非線形な制御則を獲得できる.

まず, 価値関数に基づいて学習を行う CPG-actor-critic モデルの学習法を導出する. 提案手法を 2 足歩行ロボットの自律的な歩行運動の獲得課題に対する計算機シミュレーションに適用し, 提案手法により安定した 2 足歩行を実現する CPG コントローラを獲得できることを示す. また, 学習によって得られた CPG コントローラの性質について報告する.

*奈良先端科学技術大学院大学 情報科学研究科 情報システム学専攻 博士論文, NAIST-IS-DT0161028, 2004 年 03 月 24 日.

次に, 方策勾配法に基づいて学習を行う CPG-actor-critic モデルの学習法を導出する. 近年, 学習の収束が保証された方策勾配法を用いた actor-critic モデルが提案された. 価値関数に基づく手法ではシステムの複雑さによる学習の困難さは克服できていないが, 方策勾配法を用いた手法ではこの困難が克服できる. 提案手法を 2 足歩行ロボットの自律的な歩行運動の獲得課題に対する計算機シミュレーションに適用し, 提案手法により安定した 2 足歩行を実現する CPG コントローラを獲得できることを示し, またパラメータの改善に対して良い性質を持つことを示す.

キーワード

強化学習, 2 足歩行, 中枢パターン生成器, 神経振動子, Actor-critic 法, 確率的方策勾配法

Reinforcement Learning method for control task using a pattern generator*

Yutaka Nakamura

Abstract

In order to control a robot that behaves in the real space, it is necessary to adapt the environment and to obtain control rule autonomously, because of the lack of the knowledge about the environment in advance or the dynamic change of the environment such as ground surface. Reinforcement learning satisfies this requirement, and can be applied to problems without prior knowledge about the environment or the dynamics of the controlled system. However, it is difficult to learn the control rule for a complex motion such as biped locomotion, because the system has strong non-linearity and large degrees of freedom, so that the control rule becomes complex and non-linear in itself.

Meanwhile, neurobiological studies have revealed that rhythmic motor patterns are controlled by neural oscillators referred to as central pattern generators (CPGs). Motivated by such a control mechanism, rhythmic movements controlled by CPGs have also been studied. In these studies, a CPG was implemented as a recurrent neural network, and the control signal was tuned by modifying weights of feedback connections from controlled system and those of mutual connections between CPG neurons.

In this thesis, as an autonomous learning framework for the CPG controller, I propose a reinforcement learning method based on a CPG-actor-critic architecture. In my method, the learning agent tunes the connection weights among CPGs instead of the complex control rule, such that the learning agent obtains a non-linear control rule by training the linear controller.

*Doctor's Thesis, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DT0161028, March 24, 2004.

First, I derive a learning scheme of the CPG-actor-critic model based on a value-based reinforcement learning. This method is applied to an automatic control problem of a biped robot. A computer simulation shows that my method successfully trains the CPG such that the biped robot walks stably, and I report the quality of the obtained CPG controller.

Next, I derive the learning scheme of the CPG-actor-critic model based on a stochastic policy gradient method. Recently, an actor-critic model based on the stochastic policy gradient method, whose convergence is guaranteed, has been proposed. A learning scheme based on the value function has not overcome the difficulty caused by the complexity of the system, but a scheme based on the stochastic policy gradient method can overcome it. We apply this method to an automatic control problem of a biped robot. A computer simulation shows that our method successfully trains the CPG such that the biped robot walks stably, and I report this learning scheme has good property of improving weight parameters.

Keywords:

reinforcement learning, biped locomotion, central pattern generator, neural oscillator, actor-critic method, stochastic policy gradient method

目 次

図 目 次

表 目 次

第1章　はじめに

1.1. 研究背景

これまでのヒト型の 2 足歩行の研究 [1] [2] に基づき, ヒト型ロボットによる実環境での 2 足歩行が実現された [3]. しかし, 現在, 実現されているロボットの運動は与えられた制御則に従ったものであり, 新しい環境に対しては新たな制御則を与える必要がある. 実環境で行動するロボットに対する制御を行う場合, 環境に対する知識が前もって得られなかったり, 地面などの環境が動的に変化するため, あらかじめ与えられた制御を行うのではなく自律的に環境に適応し, 制御則を獲得する枠組みが望ましい. 強化学習法 [4] はこのような要求を満たす手法で, 環境や制御対象のシステムなどの先見的な知識が無い場合にも適用できる. しかしながら, 歩行などの複雑な運動に対する制御則の学習は, システムの自由度が大きく非線形性が非常に強いことや, 制御則自体が非線形で複雑となることから困難なものとなる.

多くの生物にとって, 歩行や遊泳などのリズム運動は生存に欠かせない基本的な運動である. このような生物のリズム運動は, 様々な環境の変化や外乱に対して柔軟に適応できるという特徴を持っており, リズム運動の制御機構は生物学的にも工学的にもこれまで幅広く研究されてきた [5]. これまでの神経生理学の研究により, 生物のリズム運動は中枢パターン生成器 (Central Pattern Generator: CPG) と呼ばれる神経振動子ネットワークによって制御されていることが明らかになってきた [6] [7] [8]. また, 足などの運動器官から CPG への感覚フィードバック信号がリズム運動を安定にする上で重要な役割を果たしていることも示唆されている. このような生物の制御機構を参考にして, 周期的な運動に対する CPG を用いた制御法の研究が行われてきた [9]. このような CPG を用いた制御手法では, 制御器となる CPG を再帰結合型ニューラルネットワークで実装し, 制御対象のシステムからのフィードバック結合やニューロン間の相互結合の重みについて適切な設計を行っている. こうした研究の一つとして, Taga ら [10][11] はヒトの下肢の筋骨格系と CPG をモデル化し, 2 足歩行のシミュレーション実験に成功した. この 2 足歩行

を実現するためには, CPG を構成する数多くのニューロン間の相互結合や感覚フィードバック信号から CPG ニューロンへの結合の重みパラメータを決定する必要がある. しかし, これらの CPG のパラメータに対する設計原理が存在しないため, 物理システムや環境が変化した場合, 適切なパラメータを求めることは難しい.

CPG を用いた制御を行う場合, ロボットや環境の厳密なモデルが不必要であるため, ロボット自身が生物のように新しい環境に対して自律的に学習し適応する場合にも有利であると考えられる. この利点を生かすためにも生物の制御機構に関するさらなる研究と, それから得られた知見を工学的に応用する研究が必要である.

CPG のパラメータの学習を行う手法として, 与えられた性能指標に従ってパラメータを更新する方法 [12], 遺伝的アルゴリズム (Genetic Algorithm: GA) を用いた研究 [13] 等がある. 特に, Ogihara ら [14] は, Taga らの用いたモデルと比べてより詳細なヒトの神経筋骨格モデルに対して, GA を用いて CPG のパラメータを決定した. この GA 法は異なるパラメータを持つ多数の個体 (ロボット) を用意し, これらの個体の適応度をエネルギー消費量や転倒するまでに移動した距離をもとに評価する. そして, 適応度の高い個体の子孫を数多く残すことにより最適化を行う手法である. この手法は 2 足歩行の進化過程をシミュレーションするには優れているが, 単独のロボットの試行錯誤的学習に応用することは難しい.

本論文の目的はヒトの赤ちゃんが成長の過程で試行錯誤的に 2 足歩行を獲得するように, ロボットにも強化学習法を用いて制御則を獲得させることである. 本論文では特に CPG を用いた制御に対して強化学習法を適用する. 強化学習法は迷路の最短経路問題やゲームなどの有限状態, 有限行動空間におけるマルコフ決定過程 (MDP) に適用され成果を上げてきた [15] [4] [16] [17] [18] [19]. また, 比較的小さな連続な状態と行動空間を持つシステムに対する制御課題に対しても成果をあげてきた [20] [21]. しかし, 多自由度のシステムの制御など状態空間や行動空間が大きく複雑なシステムに対する制御課題においては, 強化学習を成功させるために効率の良い関数近似 [22] [23] [24] や階層化 [25] 等の問題に応じた工夫が必要である. 特に, 本研究で制御の対象とする 2 足歩行ロボットは不安定で, かつ高次元の連続状態, 行動空間を持った動的システムであるため, 2 足歩行運動に対する強化学習は困難と考えられる [26]. さらに, CPG を用いた制御課題に強化学習法を適用する場合, TD 学習や Q 学習, actor-critic モデルなどの標準的な強化学習法 [4] は再帰結合型ニューラルネットワークの一種である CPG コントローラの学習には適していない. このような問題点を解決するために. 本論文では CPG-actor-critic モデルと

呼ばれる新しい強化学習法を提案する.

1.2. 論文の構成

2 章では CPG を用いた制御手法の詳細と Taga ら [11] が用いた 2 足歩行ロボットシミュレータについて説明する.

3 章では価値関数に基づいた CPG-actor-critic モデルの学習法について示す. また, 提案手法を Taga ら [10] が用いた 2 足歩行ロボットシミュレータに対する CPG コントローラの自律的な獲得課題に提案手法を適用し, シミュレーションの結果, 提案手法によって安定な 2 足歩行を実現する CPG コントローラを獲得できることを示す. また, 地面のモデルが変化した場合について, 学習で得られた CPG コントローラと文献 [10] で用いられた CPG コントローラの性能の比較を行う.

4 章では方策勾配法に基づいた CPG-actor-critic モデルの学習法について示す. また, 3 章と同様に提案手法を Taga ら [10] が用いた 2 足歩行ロボットシミュレータに対する CPG コントローラの自律的な獲得課題に提案手法を適用し, シミュレーションの結果, 提案手法によって安定な 2 足歩行を実現する CPG コントローラを獲得できることを示す. また, この手法により, 価値関数に基づく学習法で問題となった学習過程の安定性について改良することができたことを示す. また, 自然方策勾配法を用いた CPG-actor-critic モデルの学習法を導出し, 2 足歩行運動の学習が通常の方策勾配法よりも高速であることを示す. さらに不整地上での学習を行い, 提案手法を用いた学習によって新しい環境に適応できることを示す.

5 章で本研究についてのまとめを行う.

第2章　パターン生成器を用いた制御法

本章では, パターン生成器を用いた制御手法について説明する.

2.1. CPG を用いた制御法

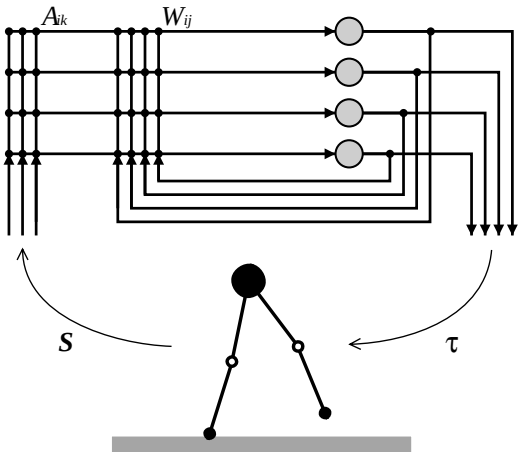


図 2.1　CPG コントローラによる制御の仕組み

ロボットのような制御対象となる物理システムの運動方程式は, 一般に

$$\dot{\mathbf{x}} = F(\mathbf{x}, \boldsymbol{\tau}) \tag{2.1}$$

と書ける. ここで, $\mathbf{x}$, $\dot{\mathbf{x}}$ はそれぞれ物理システムの状態とその時間微分で, $\boldsymbol{\tau}$ は制御器から出力される制御対象への制御信号 (トルク) である. また, $F(\mathbf{x}, \boldsymbol{\tau})$ は物理システムのダイナミクスを記述するベクトル場である.

本論文では, 図 2.1 に示すように CPG コントローラによって物理システムの制御を行うシステムを考える. CPG コントローラは神経振動子ネットワークとして実装され, 神経振動子ネットワーク内のニューロンの出力に対応した制御信号を出力する. 神経振動

子ネットワークは相互に結合した複数のニューロンで構成される再帰結合型ニューラルネットワークで, ネットワーク内の i 番目のニューロンのダイナミクスは

$$\begin{aligned} c_i\dot{\nu}_i &= -\nu_i + I_i \\ y_i &= G_i(\nu_i) \end{aligned} \tag{2.2}$$

で与えられる. ここで, ν_i, y_i, I_i, c_i はそれぞれニューロン i の内部状態, 出力, 入力と時定数である. ニューロン i の出力 y_i はニューロン i の内部状態 ν_i を出力関数 G_i で変換したもので, 出力関数 G_i としては通常, シグモイド関数やしきい値関数が使われる. また, ニューロン i への入力 I_i は

$$I_i = \sum_j W_{ij} y_j + I_i^{ext} + B_i \tag{2.3}$$

とする. ここで, 第 1 項は他のニューロンからのフィードバック入力項, 第 2 項 I_i^{ext} は感覚フィードバック信号による外部入力項, 第 3 項 B_i はバイアス入力項を表わす. W_{ij} はニューロン j から i への結合重みである. また, ニューロン i への外部入力 I_i^{ext} は

$$I_i^{ext} = \sum_k W_{ik}^{feed} X_k \tag{2.4}$$

であり, 物理システムからの感覚フィードバック信号 $\mathbf{X}$ の重み付き線形和として与えられる. ここで, W_{ik}^{feed} は感覚フィードバック信号からニューロン i への結合重みである. また, 感覚フィードバック信号 $\mathbf{X}$ は物理システムの状態 $\mathbf{x}$ の関数として与えられるベクトルで, X_k は感覚フィードバック信号の k 番目の要素である.

物理システムに対する制御信号は, CPG ニューロン出力の重み付き和で

$$\tau_n = \sum_i T_{ni} y_i \tag{2.5}$$

のように与えられる. ここで, τ_n は n 番目の制御信号で, T_{ni} は重みである.

2.2. 2 足歩行ロボットシミュレータ

本論文で学習の対象として用いた 2 足歩行ロボットシミュレータについて説明する. 2 足歩行ロボットとそれを制御する神経振動子ネットワークとして Taga ら [10] のモデルを用いた. 詳細は付録 A に示す.

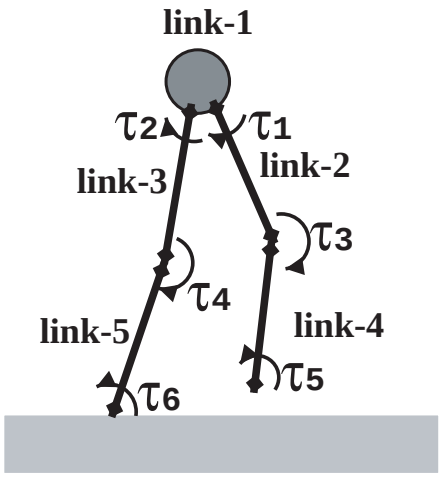
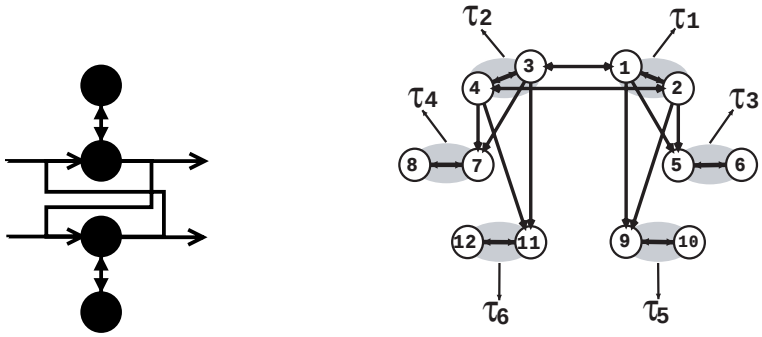


図 2.2 2 足歩行ロボットシミュレータ

2.2.1 2 足歩行ロボット

2 足歩行ロボットは図 2.2 に示すように, 互いに連結された 5 個の剛体リンクから出来ている. これらのリンクの運動はサジタル (矢状) 平面内に制限される. link-1 は質点で, 上半身を代表する. 脚は太股 (link-2, 3) と脛 (link-4, 5) で構成されており, 股, 膝, かかとの 3 つの結合部がある. 太股, 脛の長さはそれぞれ 0.5m, 0.6m である. ロボットは各結合部に加わるトルク τ_1〜τ_6 によって制御される. かかとに対するトルク τ_5, τ_6 はかかとが接地している場合だけに生成される. また, 地面からの作用はバネ・ダンパモデルとしてモデル化し, かかとの着地点からの変動量とかかとの速度に応じた反力が生じるものとする. 2 足歩行ロボットの状態 $\mathbf{x}$ は link-1 の水平と垂直位置, link-i $(i = 2, \ldots, 5)$ の鉛直軸に対する回転角とそれらの時間微分の 12 次元ベクトルである.

2 足歩行運動を制御するための CPG コントローラは再帰結合型ニューラルネットワークの一種である神経振動子ネットワークで実装される. 神経振動子ネットワークは, 12 個の主ニューロン $(i|i = 1, \ldots, 12$, 出力関数は $G_i(\nu_i) = \max(0, \nu_i))$ と, それらに付随して, 各々が対応する主ニューロンだけと結合する 12 個の副ニューロン $(i|i = 13, \ldots, 24$, 出力関数は恒等関数) の計 24 個のニューロンで構成された再帰結合型ニューラルネットワークである. このネットワーク内の相互結合の重み $\mathbf{W}^{fix}$ (式 (3.6)) は文献 [10] で用いられた値を使う. 感覚フィードバック信号が無い時, 基本 CPG 内のニューロンは自発的に周期的な信号を出力する. 2 個の主ニューロンと 2 個の副ニューロン $(2i - 1$, $2i$, $2i + 11$, $2i + 12$, $i = 1, \ldots, 6)$ が一つの神経振動子として振る舞い, 一つの神経振動子が一つのト



(a) 神経振動子　　(b) 神経振動子ネットワーク

図 2.3 CPG コントローラを構成する神経振動子ネットワーク

ルクを制御する.

2 足歩行ロボットを制御するトルク $\boldsymbol{\tau}$ は, それぞれ対応する神経振動子の主ニューロンの出力の重み付き和で生成し,

$$\tau_i = -T_i^F y_{2i-1} + T_i^E y_{2i} \qquad i = 1, \ldots, 4$$
$$\tau_i = (-T_i^F y_{2i-1} + T_i^E y_{2i})\Xi_{i-1} \qquad i = 5, 6$$

とする. ここで, Ξ_{i-1} はかかとの指標関数で, link-i のかかとが接地している時に 1, そうでないときに 0 である. τ_1, τ_2 が股関節, τ_3, τ_4 が膝関節, τ_5, τ_6 がかかとを制御するトルクである. T_i^F と T_i^E はそれぞれ屈筋と伸筋に加わる力を決める重みで, 3 章, 4 章の 2 足歩行シミュレーション実験では文献 [10] で用いられた値を使う.

文献 [10] で用いられた 2 足歩行ロボットからの感覚フィードバック信号は $\mathbf{X} = \{x_3, x_4, x_5\Xi_4, x_6\Xi_5, \Xi_4, \Xi_5, x_{11}\Xi_4, x_{12}\Xi_5\}$ である. ここで, $x_i, i = 3, \ldots, 6$ は link-$(i-1)$ の角度で, x_{11} と x_{12} はそれぞれ link-4 と link-5 の角速度である. 3 章, 4 章の 2 足歩行シミュレーション実験では, 感覚フィードバック信号として $\mathbf{X}$ を用いる.

第3章　CPG-actor-critic モデル

本章では, CPG コントローラを用いた制御法に対して自律的な学習を行う枠組みとして, CPG-actor-critic モデルと呼ばれる新しい強化学習法を提案する.

2 足歩行ロボットの自律的な歩行運動の獲得課題に対する計算機シミュレーションを行い, 提案手法により安定した 2 足歩行を実現する CPG コントローラを獲得できることを示す. また, 学習によって得られた CPG コントローラの性質について述べる.

3.1. 強化学習

制御対象のシステムが状態空間 $\mathcal{S}$, 行動空間 $\mathcal{U}$ を持つマルコフ決定過程 (MDP) でモデル化できるとする. 時刻 t において, 学習システムは制御対象の状態 $\mathbf{s}(t) \in \mathcal{S}$ を観測し, 時刻に依存しない方策 $\pi(\cdot)$ に従い制御信号 $\mathbf{u}(t) \in \mathcal{U}$ を出力する. この制御信号は

$$\mathbf{u}(t) = \pi(\mathbf{s}(t))$$

のように計算される. 制御信号 $\mathbf{u}(t)$ を受け取り, 制御対象のシステムの状態はシステムのダイナミクスに従い, $\mathbf{s}(t)$ から $\mathbf{s}(t+1)$ へ状態遷移する. 同時に学習システムには即時報酬 $r(\mathbf{s}(t), \mathbf{u}(t))$ が与えられる. 本章で定式化する強化学習の目的は

$$V^{\pi}(\mathbf{s}(t)) \equiv \mathrm{E}_{\pi}\left[\sum_{\tau=0}^{\infty} \gamma^{\tau-t} r(\mathbf{s}(t+\tau), \mathbf{u}(t+\tau))\right] \tag{3.1}$$

と定義される期待累積報酬 (価値関数) を最大化する方策 $\pi(\cdot)$ を求めることである. ここで, E_{π} は方策 $\pi(\cdot)$ に従い制御信号を出力する場合の状態と行動の系列に関する期待値である. また, $\gamma \in (0, 1]$ は減衰係数である. $V^{\pi}(\cdot)$ は状態価値関数 (評価関数) と呼ばれる状態 $\mathbf{s}$ から累積報酬 R の期待値への写像で, 現在の方策 π に依存する. また, Q 関数とも呼ばれる行動評価関数 $Q(\mathbf{s}, \mathbf{u})$ は,

$$Q^{\pi}(\mathbf{s}(t), \mathbf{u}) = r(\mathbf{s}(t), \mathbf{u}) + \gamma V(\mathbf{s}(t+1)) \tag{3.2}$$

と定義される．行動評価関数 $Q^{\pi}(\mathbf{s}, \mathbf{u})$ は, 時刻 t において行動 $\mathbf{u}$ を選択し, 時刻 $t+1$ 以後において方策 π に従って制御を行った場合に学習システムが獲得する期待累積報酬を表す．式 (3.1) と式 (3.2) から, 行動評価関数はベルマン方程式と呼ばれる自己無撞着等式:

$$Q^{\pi}(\mathbf{s}(t), \mathbf{u}(t)) = r(\mathbf{s}(t), \mathbf{u}(t)) + \gamma Q^{\pi}(\mathbf{s}(t+1), \bar{\mathbf{u}}) \tag{3.3}$$

を満たす必要がある．ここで, $\bar{\mathbf{u}}$ は制御器による時刻 $t+1$ での制御信号であり, $\mathbf{u}(t+1)$ を用いることができる．

3.1.1 Actor-critic モデル

本章で提案する手法は, 強化学習でしばしば用いられる actor-critic モデル [4] [22] [23] に基づくものである．Actor は制御対象に対して制御信号を生成する制御器で, 物理システムの状態に対応した制御信号を出力する．一方 critic は, 現在の actor を用いて物理システムを制御した場合, 将来にわたって得られる報酬の和 (価値関数) を予測する．そして, actor は critic が予測する価値関数の値をより大きくするように更新される．

提案手法である CPG-actor-critic モデルではSARSA アルゴリズム [4] と同様に, critic は式 (3.3) を満たすように更新され, 現在の actor に依存した行動評価関数を近似する．

3.2. 提案モデル

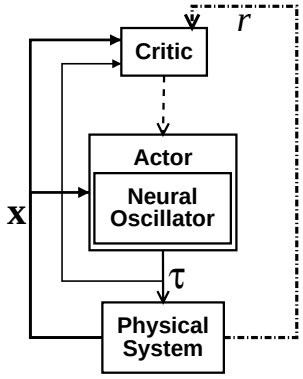


図 3.1 Actor-critic モデル

3.1.1 節で説明した actor-critic モデルを図 3.1 に示すように, CPG コントローラを用

いた制御課題に適用しようとする場合, いくつかの問題が生じる. CPG コントローラは再帰結合型ニューラルネットワークである神経振動子ネットワークで実装されているが, actor-critic モデルは時間的に局所的な誤差 (TD 誤差) に基づいて学習を行うため, 再帰結合型ニューラルネットワークの学習には適していない.

また, 物理システムのダイナミクスは運動方程式 (2.1) で表され, 物理システムの状態 $\mathbf{x}$ と制御トルク $\boldsymbol{\tau}$ に依存する. ところが, CPG コントローラを用いて制御を行う場合, $\boldsymbol{\tau}$ は CPG ニューロンの内部状態 $\boldsymbol{\nu}$ に依存するため, システム全体のダイナミクスは

$$(\dot{\mathbf{x}}, \dot{\boldsymbol{\nu}}) = F_{\text{CPG-coupled system}}(\mathbf{x}, \boldsymbol{\nu}, \boldsymbol{\tau}, \mathbf{I}) \tag{3.4}$$

となっている. よって, 本来は物理システムの状態 $\mathbf{x}$ と CPG ニューロンの内部状態 $\boldsymbol{\nu}$ に依存する式 (3.4) のダイナミクスを持つシステムを物理システムの状態 $\mathbf{x}$ だけに依存する式 (2.1) のダイナミクスを持つシステムであると見なすと, システムが隠れた状態変数 $\boldsymbol{\nu}$ を持つ事になる. 通常, 強化学習では全ての状態変数を観測できるマルコフ決定過程 (MDP) の問題を扱うが, ダイナミクスに隠れた状態変数を持つ際の最適制御問題は, 部分観測マルコフ決定過程 (POMDP) と呼ばれ, 隠れた状態変数を持たない MDP に比べてはるかに難しい問題となる. 実際に, POMDP に対して隠れた状態変数を無視して強化学習法を適用することは, 有効ではないことが知られている [27]. CPG コントローラによる学習を MDP として扱うためには, 式 (3.4) のように物理システムと神経振動子ネットワーク内のニューロンを一つのシステムとして扱い (以後 CPG 結合システムと呼ぶ), その CPG 結合システムを actor によって制御するモデルへと変更することが有効である.

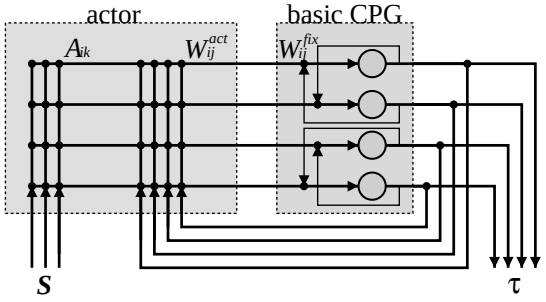


図 3.2 Actor と基本 CPG

そこで, 我々の提案手法では図 3.2 に示すように CPG コントローラを基本 CPG と actor の 2 つのモジュールに分割する. 基本 CPG は結合重み W_{ij}^{fix} を持つ神経振動子ネッ

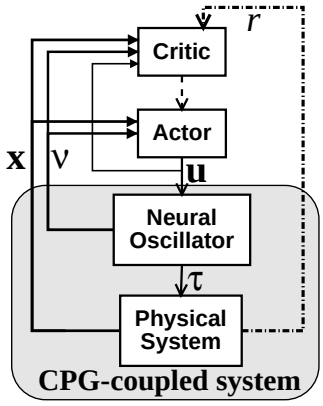


図 3.3 CPG-actor-critic モデル

トワークである. また, actor は基本 CPG ニューロンの出力と感覚フィードバック信号を入力信号とし, これに応じて制御信号 (以後間接制御信号と呼ぶ) を基本 CPG に対して出力する. このモジュール分割によって, CPG ニューロンへの入力 I_i (式 (2.3)) は以下のように 2 分割される.

$$I_i = I_i^{fix} + u_i \tag{3.5}$$

$$I_i^{fix} = \sum_j W_{ij}^{fix} y_j + B_i \tag{3.6}$$

$$u_i = \sum_j W_{ij}^{act} y_j + \sum_k W_{ik}^{feed} X_k \tag{3.7}$$

ここで, I_i^{fix} は基本 CPG 内の相互結合による入力とバイアス入力を合わせた入力で, W_{ij}^{fix} や $\mathbf{B}$ の値は変化しないものとする. また, $\mathbf{u}$ は actor の出力する間接制御信号で, actor は基本 CPG ニューロンの出力 $\mathbf{y}$ と感覚フィードバック信号 $\mathbf{X}$ を入力とした線形制御器となっている. Actor のパラメータである結合重み W_{ij}^{act} と W_{ik}^{feed} は可変なパラメータで, 後述するように強化学習法を用いて学習する. 物理システムへの制御トルクは式 (2.5) で与えられ, この結合重みの値 $\mathbf{T}$ は変化しないものとする. 以上の構造を持つモデルを CPG-actor-critic モデルと呼ぶ.

この CPG-actor-critic モデルに対しては 2 通りの見方ができる. まず, 制御の観点からは, 図 2.1 に示すように基本 CPG と actor を合わせた, すなわち結合重み $W_{ij} = W_{ij}^{fix} + W_{ij}^{act}$ を持つ神経振動子ネットワークとして構成された CPG コントローラによって物理システムが制御されるというモデルである. 一方, 強化学習の観点からは, 図 3.3 に示すように, 基本 CPG と物理システムの二つを合わせて一つのシステムと見なした

CPG 結合システムを, actor が間接制御信号 $\mathbf{u}$ によって制御するというモデルである. このモデルにおいては, actor は線形制御器となっており相互フィードバック結合を持たない. このため, 時間逆向きのバックプロパゲーション学習が必要な再帰結合型ニューラルネットの学習 [28] [29] を行わずに, CPG コントローラの結合重みパラメータの学習ができる. Actor のパラメータ W_{ij}^{act} と W_{ik}^{feed} は例えば後述する確率勾配法 [30] によって学習することができる. このモデルをより一般化した非線形結合を持つ actor を仮定しても, actor が相互フィードバック結合を持たないため, 確率勾配法による学習が可能である.

また, critic は基本 CPG の内部状態 $\boldsymbol{\nu}$ と物理システムの状態 $\mathbf{x}$ を合わせた CPG 結合システムの状態 $(\boldsymbol{\nu}, \mathbf{x})$ を観測し, 将来にわたって得られる報酬の和 (価値関数) を予測する. このような CPG-actor-critic モデルを用いた強化学習では, 物理システムと基本 CPG の全ての状態を観測することができれば, 価値関数の推定に必要な情報を全て観測できるため, 前述の POMDP の困難さを回避できる.

3.2.1 学習アルゴリズム

本節では, CPG-actor-critic モデルに対する強化学習アルゴリズムを説明する. 簡単のため, 微分方程式 (2.1) と (2.2) に対して適当な方法で離散化を行い, 離散時間の問題として扱う.

時刻 t において, actor は基本 CPG ニューロンの出力 $\mathbf{y}(t)$ と感覚フィードバック信号 $\mathbf{S}(t)$ を観測し, 式 (3.7) に従い間接制御信号 $\mathbf{u}(t)$ を出力する. CPG 結合システムは式 (2.1)〜(3.6) に従い, 状態を $(\boldsymbol{\nu}(t), \mathbf{x}(t))$ から $(\boldsymbol{\nu}(t+1), \mathbf{x}(t+1))$ に変化させる. この時, critic はシステムの状態の瞬時的良さに応じて直接報酬 $r(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u}(t))$ を受け取るものと仮定する.

行動評価関数

通常の actor-critic モデルにおける価値関数 (式 (3.1)) と行動評価関数 (式 (3.2)) は, CPG-actor-critic モデルではそれぞれ

$$V(\boldsymbol{\nu}, \mathbf{x}) \equiv \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u}(t))\right] \tag{3.8}$$

$$Q(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u}) = r(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u})\gamma V(\boldsymbol{\nu}(t+1), \mathbf{x}(t+1)) \tag{3.9}$$

と書き換えられる. これにより, ベルマン方程式は

$$Q(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u}(t)) = r(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u}(t)) + \gamma Q(\boldsymbol{\nu}(t+1), \mathbf{x}(t+1), \bar{\mathbf{u}}) \tag{3.10}$$

となる. 提案手法である CPG-actor-critic モデルでは SARSA アルゴリズム [4] と同様に, critic は式 (3.10) を満たすように更新され, 現在の actor に依存した行動評価関数を近似する.

正規化ガウス関数ネットワーク (NGnet)

Critic のための関数近似器として, 正規化ガウス関数ネットワーク (NGnet) [31] [32] を用いる. N 次元ベクトル $\mathbf{S} = (\boldsymbol{\nu}, \mathbf{x}, \mathbf{u})$ を入力とする NGnet は以下のように定義される.

$$q = Q(\mathbf{S}) = \sum_{m=1}^{M} \left(\frac{\mathcal{N}_m(\mathbf{S})}{\sum_{j=1}^{M} \mathcal{N}_j(\mathbf{S})} \right) q_m \tag{3.11}$$

$$q_m = \mathbf{K}_m \cdot \mathbf{S} + b_m \tag{3.12}$$

$$\mathcal{N}_m(\mathbf{S}) \equiv (2\pi)^{-N/2} |\boldsymbol{\Sigma}_m|^{-1/2} \exp\left[-\frac{(\mathbf{S}-\boldsymbol{\mu}_m)' \boldsymbol{\Sigma}_m^{-1} (\mathbf{S}-\boldsymbol{\mu}_m)}{2} \right]$$

ここで $\mathbf{S}'$ は $\mathbf{S}$ の転置を示す. M は NGnet のユニット数で, $|\cdot|$ は行列式を示す. $\mathcal{N}_m(\mathbf{S})$ は平均 $\boldsymbol{\mu}_m$, 共分散行列 $\boldsymbol{\Sigma}_m$ の N 次元ガウス関数で, q_m と $\mathbf{K}_m$, b_m はそれぞれ, ユニット m の出力と線形回帰行列, バイアス項である. NGnet は入力と出力の組 $(\mathbf{S}, q)$ を確率事象とした確率モデルとして定式化できる. 各事象に対して一つのユニットが選択されると仮定し, 選択されたユニット番号 m を隠れ変数として扱う. $(\mathbf{S}, q, m)$ を完全事象と呼び, 確率モデルは完全事象に対する以下の確率分布によって定義される.

$$p(\mathbf{S}, q, m|\theta) = M^{-1} \mathcal{N}_m(\mathbf{S}) (2\pi)^{-\frac{1}{2}} \sigma_m^{-1} \exp\left[-\frac{(q-q_m)^2}{2\sigma_m^2} \right] \tag{3.13}$$

ここで, $\theta = \{\boldsymbol{\mu}_m,\ \boldsymbol{\Sigma}_m,\ \sigma_m,\ \mathbf{K}_m | m = 1, \ldots, M\}$ はモデルパラメータである. また, σ_m は出力 q の標準偏差である. この分布から $\mathbf{S}$ が与えられた場合の出力 q の期待値 $E[q|\mathbf{S}] \equiv \int qP(q|\mathbf{S})dq$ を計算すると, NGnet(式 (3.11)) の出力と一致することが分かる. すなわち, 確率分布 (3.13) は式 (3.11) で定義された NGnet の確率モデルとなっている.

観測されたデータに基づいて, 確率モデル (3.13) のモデルパラメータ θ を最尤推定法により推定する. 特に隠れ変数をもつ確率モデルに対しては EM アルゴリズム [33] が適

用できる. NGnet に対するオンライン EM アルゴリズムは動的な環境に対しても効果的であることが示された [31]. CPG-actor-critic モデルでは, critic が予測する行動評価関数は actor に依存しており, また actor も critic の予測に従って学習を行うため, critic の学習する目標関数が学習の進行と共に変化する. 高次元空間上で関数近似を行う場合, 空間全体にユニットを配置すると多数のユニットが必要となる. ユニット数の増加と共に計算量的も大きくなるが, 強化学習問題ではしばしば入力空間内の一部の部分空間に偏ってデータが出現するために入力空間全体にユニットを配置することは無駄であり, NGnet ではデータの出現頻度に基づいてユニットを配置する. オンライン EM アルゴリズム [31] にはユニットの生成・削除機構が組み込まれておりこのように対象が動的に変化する場合にも有効である. そこで, critic の学習はオンライン EM アルゴリズムにより行う. 学習アルゴリズムの詳細は, 付録 B に示す.

また, CPG-actor-critic モデルを 2 足歩行ロボットシミュレータの学習課題に適用する場合, 入力次元が 29 次元となり, 非常に高次元の関数近似を行うことになるため, NGnet の学習方法に以下の変更を加えた.

- NGnet はデータの出現頻度に基づいてユニットを配置する. よって, 出現頻度の非常に低い入力 $\mathbf{S}$ に対しては NGnet の出力は信頼できない. そこで, 全てのユニット m について $p(\mathbf{S}|m,\theta)$ が小さい場合, 式 (3.10) の右辺で与えられる出力の教師信号 $\hat{q}$ の状態 $\{\mathbf{x},\boldsymbol{\nu}\}$ と行動 $\mathbf{u}$ に関する期待値を出力するように変更する. 付録 B.1

- $p(\mathbf{S}|m,\theta) >> 0$ となる領域が小さい場合, 線形回帰行列 $\mathbf{K}_m$ の推定精度が悪くなるため, 線形回帰行列 $\mathbf{K}_m$ を 0 にする. 付録 B.2

- NGnet は局所モデルであり, $p(\mathbf{S}|m,\theta) << 1$ となる入力 $\mathbf{S}$ が与えられた場合, ユニット m の出力 q_m は NGnet の出力 q にほとんど影響を与えない. 一方, 学習時に $p(\mathbf{S},\hat{q}|m,\theta) << 1$ となるデータが観測されると, ユニット m のモデルパラメータを推定するために保持している十分統計量が小さくなり, 数値誤差の影響が大きくなる. よって, 新規に学習データ $\{\mathbf{S},\hat{q}\}$ に対して $p(m|\mathbf{S},\hat{q},\theta) << 1$ となる場合, ユニット m に対する学習を行わない. これにより, 離れたデータによる影響が無くし, さらに計算量を削減できる. 付録 B.3

3.2.2 エピソード学習

CPG-actor-critc モデルにおける学習は以下のような過程で行う.

Critic 学習ステップ　時刻 t において, actor が式 (3.7) に従い間接制御信号 $\mathbf{u}(t)$ を出力する．この後, CPG 結合システムは式 (2.1)〜(3.6) に従い状態を $(\boldsymbol{\nu}(t+1), \mathbf{x}(t+1))$ に変化させる．同時に, critic は直接報酬 $r(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u}(t))$ を観測し, 式 (3.10) の両辺の差 (TD 誤差) を減少させるようにオンライン方式で学習を行う．この時, critic の入力は $(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u}(t))$ であり, 出力の教師信号は式 (3.10) の右辺で与えられる．この入力と出力教師信号を用いて, NGnet のパラメータをオンライン EM アルゴリズムにより学習更新する．

学習システムは一定時間 (t_{max}) が経過するか, 物理システムが終了状態になるまで, 上記の actor による制御と critic の学習を繰り返す．すなわち, この間は actor のパラメータを固定している．また, 制御を行っている間の CPG 結合システムの状態の系列 $\{\boldsymbol{\nu}(t), \mathbf{x}(t) | t = 0, 1, \ldots, t_{max}\}$ は次ステップの actor の学習のために記録される．

Actor 学習ステップ　次に, 記録された各時刻の状態 $(\boldsymbol{\nu}(t), \mathbf{x}(t)), t = 0, 1, \ldots, t_{max}$ に対して行動評価関数の値を増加させるように actor のパラメータを更新する．Actor の結合重みは勾配法を用いて

$$\hat{W} := \hat{W} + \alpha \frac{\partial Q(\boldsymbol{\nu}, \mathbf{x}, \mathbf{u})}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \hat{W}} \tag{3.14}$$

のように更新される [34].

ここで, $\hat{W}$ は actor の結合重み W_{ij}^{act} と W_{ik}^{feed} を表しており, α は学習係数で正の定数とする．また, $\mathbf{u}$ は CPG 結合システムの状態が $(\boldsymbol{\nu}, \mathbf{x})$ の場合の actor の出力で, 式 (3.7) で計算できる．行動評価関数の微分値 $\partial Q(\boldsymbol{\nu}, \mathbf{x}, \mathbf{u})/\partial \mathbf{u}$ の計算には現在の critic (式 (3.11)) を用い, $\partial \mathbf{u}/\partial \hat{W}$ の計算には式 (3.7) を用いる．このパラメータ更新は保存された軌道に従って疑似オンライン的に行う．すなわち, actor は時々刻々と変化する各状態に対して critic の予測する期待報酬が大きくなるようにパラメータを更新する．

以上の 2 つの学習ステップを 1 エピソードとし, 強化学習はこのエピソードを繰り返すことで行われる．本手法では actor と critic に対して同時にオンライン学習を行うことは可能である．しかし, actor の学習によって critic の学習目標が変化し, critic の学習によって actor の学習目標が変化するという相互に依存した関係があるため, actor 学習時には critic を固定し, critic 学習時には actor を固定することで学習が安定する．本手法ではこの安定性を実現する一つの方法としてエピソードごとに critic の学習と actor の学習を交互に行う方法を用いる．

3.3. 実験

CPG-actor-critic モデルに基づく強化学習法を 2 足歩行ロボットシミュレータ [10] に適用した. シミュレーション実験の目的は, ロボットが安定した歩行運動を行うような CPG コントローラのパラメータを自律的に獲得することである.

3.3.1 実験条件

CPG 結合システムのダイナミクスは時間間隔 0.0001 秒の Runge-Kutta 法を用いた積分によって計算した. また, 学習システムは 0.01 秒毎にシステムの状態を観測し, 制御信号を出力した. また, 各リンクの角速度の観測値はサンプリング時刻における値ではなく, 0.01 秒間の平均を用いた.

基本 CPG として文献 [10] で用いられた神経振動子ネットワークを用いた. また, 今回の実験では, 学習を容易にするために, actor のパラメータの内, 学習できるパラメータを制限した. ニューロン間の相互結合の重みパラメータを固定, すなわち $W_{ij}^{act} \equiv 0$ とした. さらに, CPG ニューロンに対する外部入力は文献 [10] で用いられたものと同様とし, 感覚フィードバック信号から CPG ニューロンへの結合には

$$
\begin{aligned}
I_1^{ext} &= a_1 X_1 - a_2 X_2 + a_3 X_3 + a_4 X_6 \\
I_3^{ext} &= a_1 X_2 - a_2 X_1 + a_3 X_4 + a_4 X_5 \\
I_5^{ext} &= a_5 X_4 \\
I_7^{ext} &= a_5 X_3 \\
I_9^{ext} &= -a_6 X_3 - a_7 X_4 - a_8 X_7 \\
I_{11}^{ext} &= -a_6 X_4 - a_7 X_3 - a_8 X_8 \\
I_{2i}^{ext} &= -I_{2i-1}^{ext} \qquad \text{for } i = 1, \ldots, 6
\end{aligned}
\tag{3.15}
$$

のように制限を加えた形で与えた. 以後の実験では, 間接制御信号を $u_i = I_{2i-1}^{ext}, i = 1, \ldots, 6$ とし, 式 (3.15) 中の結合重み $\{a_i | i = 1, \ldots, 8\}$ だけを学習対象とした.

直接報酬 $r(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u}(t))$ は次時刻のロボットの状態 $\mathbf{x}(t+1)$ だけに依存するものと

して, $\tilde{r}(\mathbf{x}(t+1))$ で与え,

$$\tilde{r}(\mathbf{x}) = k_h r_h(\mathbf{x}) + k_s r_s(\mathbf{x}) \tag{3.16}$$
$$r_h(\mathbf{x}) = h_1 - \min(h_4, h_5) - H$$
$$r_s(\mathbf{x}) = \begin{cases} x_7 & \text{if } |x_7| < 1 \\ x_7/|x_7| & \text{otherwise} \end{cases}$$

のように定義した. ここで, h_i $(i = 4, 5)$ は link-i のかかとの高さで, x_7 は link-1 の水平方向に対する速度を示している. 報酬 $r_h(\mathbf{x})$ は腰の位置が高い程大きくなり, ロボットが転倒しないように学習するのに役立つ. 報酬 $r_s(\mathbf{x})$ は右方向への速度が大きいほど大きくなる. この報酬項は, ロボットが前方向に歩くのを促進する働きがある. また, k_h と k_s はそれぞれ r_h と r_s に対する重みで H はしきい値である. 実験では, 重み k_h と k_s をそれぞれ 0.5 と 0.02 とし, しきい値 H を 0.8 とした.

1 エピソードは最大 5 秒間とし, 5 秒以内にロボットが転倒した場合はその時点でエピソードを終了させる. 学習の初期段階において, 各エピソードのロボットの初期状態は脚をわずかに開いて静止した状態, ニューロンの初期状態は全てのニューロンの内部状態を 0 とした. また, 初期状態の脚の角度はエピソード毎に一定の角度内でランダムに異なる角度を選んだ. 学習が進行し 5 秒間ロボットが転倒しないエピソードが起った場合, それ以後のエピソードでは, 転倒しなかったエピソードのランダムに選んだ時刻における CPG 結合システムの状態をエピソード開始時の初期状態とした.

学習のはじめに, actor のパラメータである結合重み $\mathbf{a}$ は小さな乱数で与えた. また, 2 足歩行ロボットの link-1 の水平位置は 2 足歩行の制御に無関係で, 価値関数に影響を与えないと考えられるため, critic の入力に含めなかった. また, 副ニューロンは主ニューロン以外との結合を持たないため, 副ニューロンの状態変数は主ニューロンの状態変数に強く依存していると考えられる. そこで, 副ニューロンの状態も critic の入力に含めなかった. さらに 4.2.1 節で説明した手続きに変更を加え, critic 学習ステップを 10 回行い, その後に actor 学習ステップを行うという手続きで行った. 強化学習はこのような学習手続きを繰り返すことによって行われた.

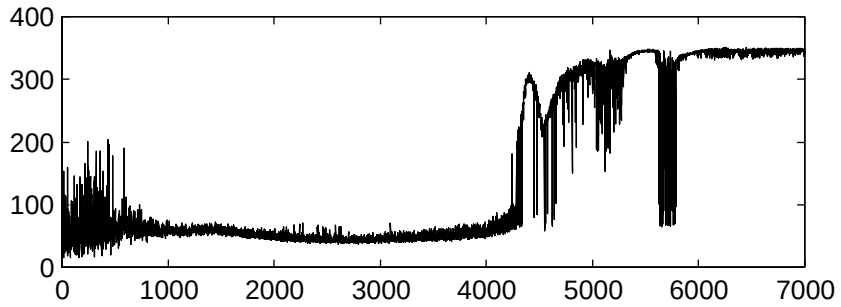
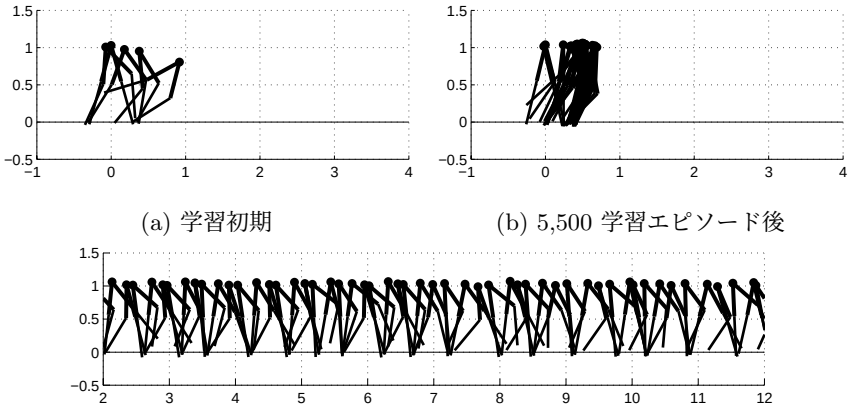


図 3.4 累積報酬



(a) 学習初期

(b) 5,500 学習エピソード後

(c) 7,000 学習エピソード後

図 3.5 ロボットの歩行パターン

3.3.2 実験結果

図 3.4 に強化学習の学習曲線を示す. 横軸は学習エピソード数を示し, 縦軸は 1 エピソード内での累積報酬を示す. 学習開始前の段階ではロボットはすぐに転倒する (図 3.5 左上). 約 4000 エピソード後にロボットは足踏みを始めるようになり, 転倒しなくなる (図 3.5 右上). 約 5800 エピソード後にロボットは安定に歩行するようになる (図 3.5 下). 学習過程における太股 (link-2) の角度 x_3 とそれに対応するニューロンの状態 ν_1 の位相図の変化を図 3.6 に示す. 各図の横軸と縦軸がそれぞれ角度とニューロンの状態である. 左上の図は文献 [10] で用いられた値を用いた場合で, その他の図は 5300 学習エピソード後 (足踏み) から 7000 学習エピソード後までの間の actor によって制御を行った場合である. 学習の進行に従って, ニューロンとロボットが一定の位相関係を満たすようになることが分かる.

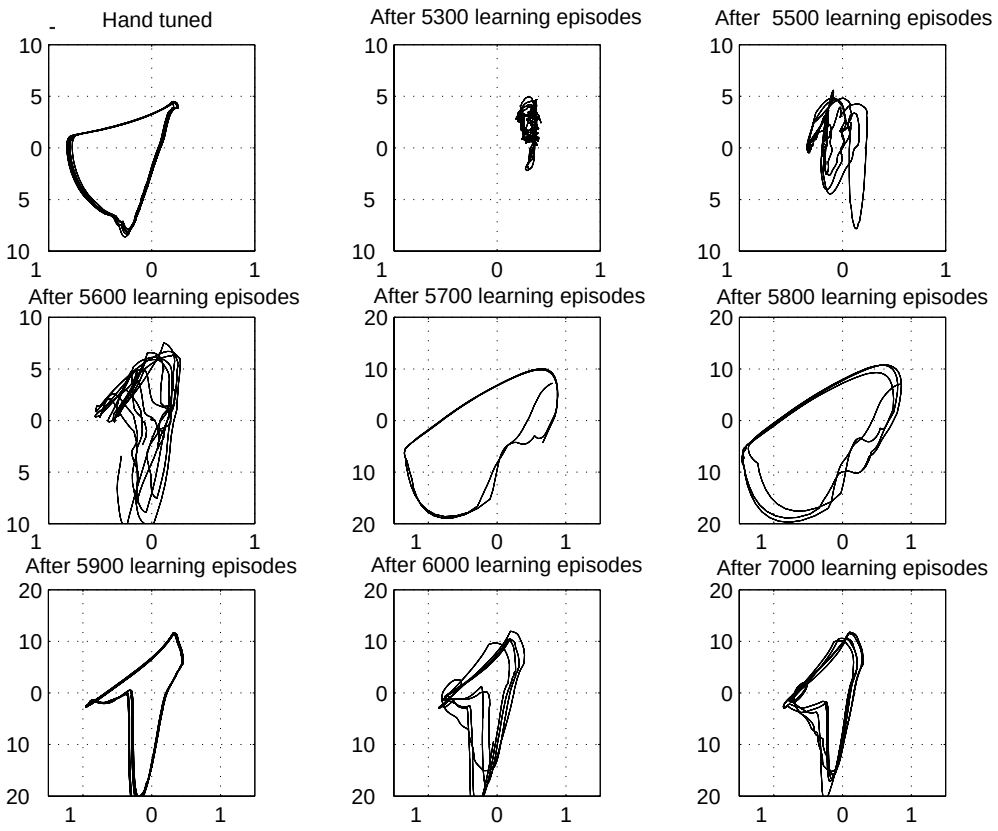
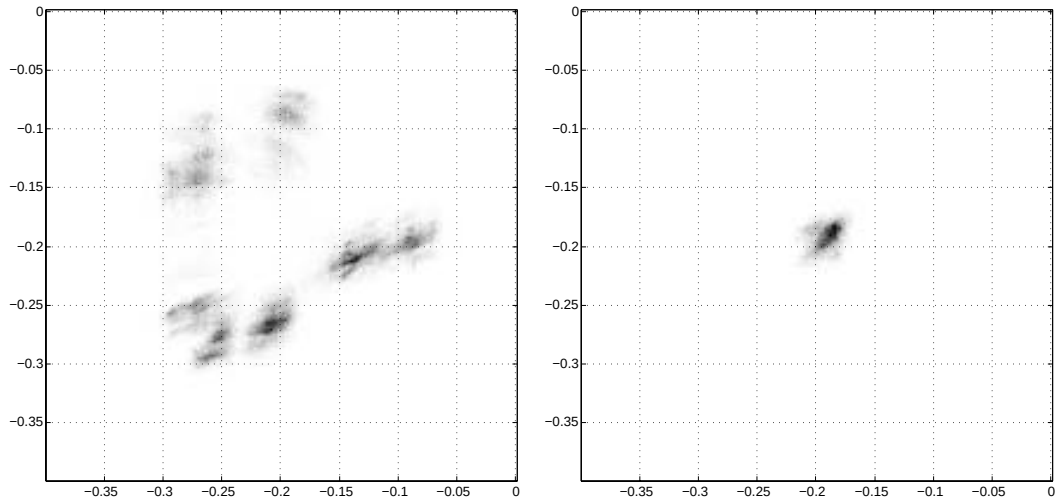


図 3.6 学習の進行に従った位相の変化

学習の結果, 得られた感覚フィードバック信号からの結合重みの値は $\mathbf{a}_{RL} = \{10.0, -3.1, 10.0, 6.5, 1.1, 2.5, 10.0, 4.9\}$ であった. これは文献 [10] で用いられた値 $\mathbf{a}_{HT} = \{1.5, 1.0, 1.5, 1.5, 3.0, 1.5, 3.0, 1.5\}$ とは大きく異なっている.

学習で得られたパラメータ $\mathbf{a}_{RL}$ と文献 [10] のパラメータ $\mathbf{a}_{HT}$ の比較

太股の角度 θ_2 の描くリターンマップを図 3.7 に示す. 右図と左図はそれぞれ $\mathbf{a}_{RL}$ と $\mathbf{a}_{HT}$ を用いてロボットを制御したときのリターンマップで, ポアンカレ切断面は $y_3 = 3.0$ とした. 図 3.8 に最大リアプノフ指数を示す. 初期状態から $\mathbf{a}_{RL}$(実線) と $\mathbf{a}_{HT}$(破線) を用いて CPG 結合システムの制御を行い基準軌道を生成し, 得られた基準軌道の各時刻において近接軌道を生成することで, 最大リアプノフ指数を計算した. ただし, 近接軌道の生成にはニューロンの状態 $\boldsymbol{\nu}$ だけを用いた. 横軸は時刻で, 縦軸は最大リアプノフ指数で



(a) 学習で得られたパラメータ $\mathbf{a}_{RL}$　　(b) 文献 [10] のパラメータ $\mathbf{a}_{HT}$

図 3.7　リターンマップ

ある．学習によって得られたパラメータ $\mathbf{a}_{RL}$ の場合のリターンマップは $\mathbf{a}_{HT}$ の場合に比べて複雑な形状をしているが, 最大リアプノフ指数は負となっている．これは, ロボットの軌道が引き込まれるアトラクタが, 外乱によって擬周期的になっていることを反映していると考えられる．2 足歩行ロボットシミュレータの物理モデルでは, 例えば, ロボットの足が着地したときや, ひざの角度が π を越えたときに衝撃力が加わるが, これらが外乱となる．

様々な地面のモデルにおいて, $\mathbf{a}_{RL}$ と $\mathbf{a}_{HT}$ を用いて制御を行った場合の歩容をそれぞれ図 3.10 と図 3.11 に示す．上り (急), 上り (緩), 下り (緩) と下り (急) の勾配はそれぞれ正接が 0.1, 0.05, -0.05 と -0.1 となる勾配で, デコボコ道は 1m 毎に正接が 0.1 と -0.1 の勾配が繰り返されるものとした．学習によって得られたパラメータにより, 坂道やデコボコ道に対して, $\mathbf{a}_{HT}$ の場合よりも安定した歩行が可能になっていることが分かる．また, 様々な地面のモデルにおける太股 θ_2, 脛 θ_4 の角度-角速度の位相図の変化を図 3.9 に示す．上段の図は $\mathbf{a}_{HT}$ によって, 下段の図は $\mathbf{a}_{RL}$ によって制御を行った場合のもので, 各図は左から平面, 登り坂, 下り坂, デコボコ道において実験した場合の位相図である．また, 横軸と縦軸はそれぞれ角度と角速度であり, 実線は θ_2, 点線は θ_4 の位相図である．学習で得られたパラメータによる制御の方が, $\mathbf{a}_{HT}$ による制御よりも位相図の変化が小さく, 歩容の変化が小さいことが分かる．

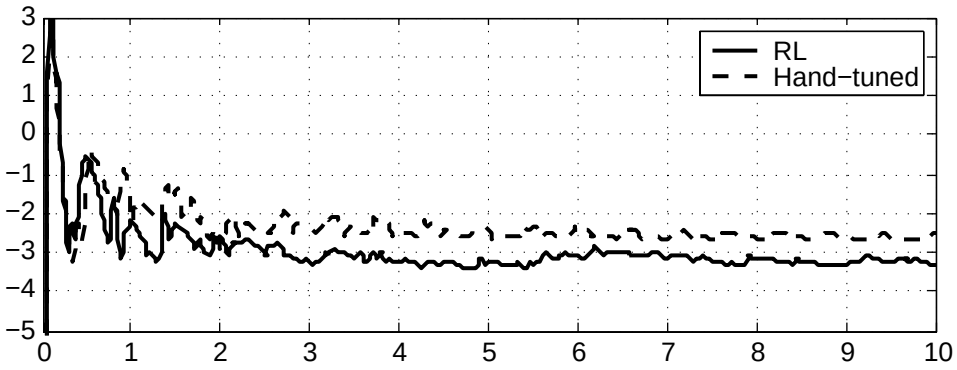


図 3.8 最大リアプノフ指数

3.4. 考察

2 足歩行ロボットの制御問題に提案手法を適用し, 強化学習に基づく 2 足歩行運動が実現できた. 学習によって得られたパラメータを用いた場合, 様々な地面のモデルにおいて安定に歩行できた. 強化学習法では, 学習時により良いパラメータを探索するためにランダム性を加える. Actor の学習に用いる保存した軌道は着地時の衝撃や制御器に加えられる外乱などの影響を受けたものとなっている. このため, 歩行時のランダムな変動を制御できるような actor が学習により獲得できたと考えられる.

一方で, 学習の過程はかなり不安定であった. 学習過程の不安定さはダイナミクスが非常に複雑であるために評価関数の学習が困難であることが理由の一つとして考えられる. このことから, 学習の安定性を高める手法について研究を進める必要が示唆される.

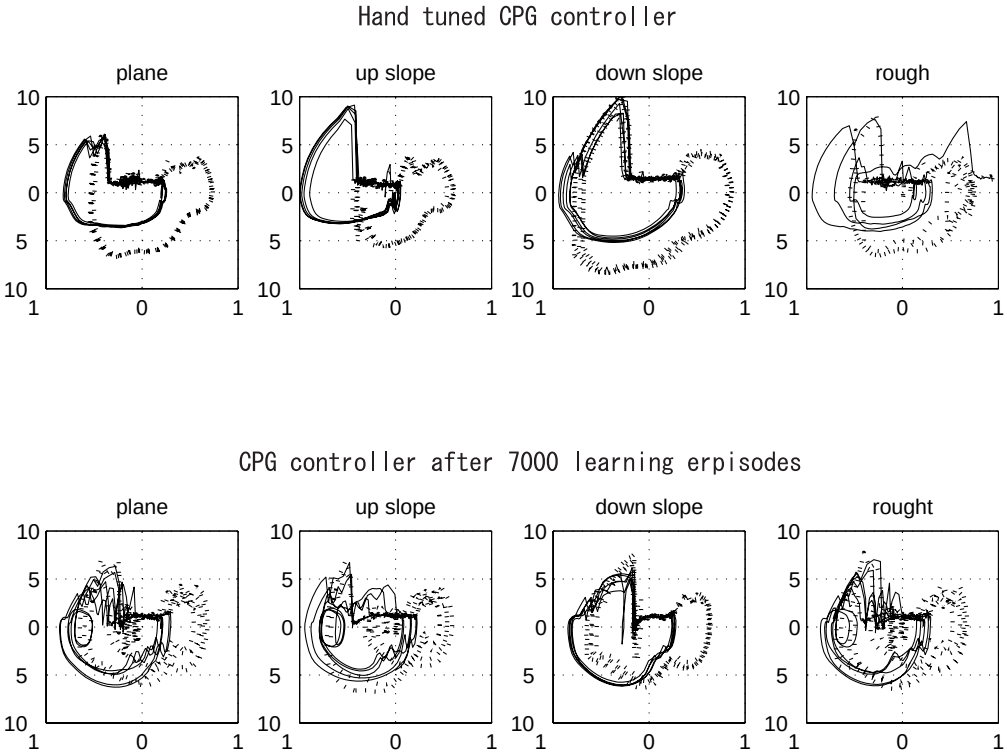
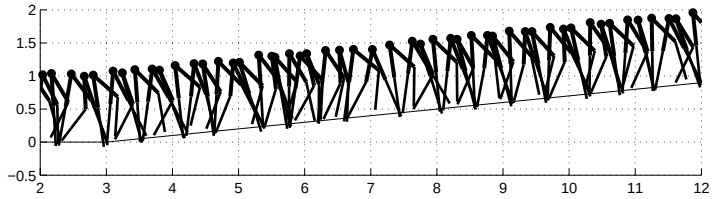
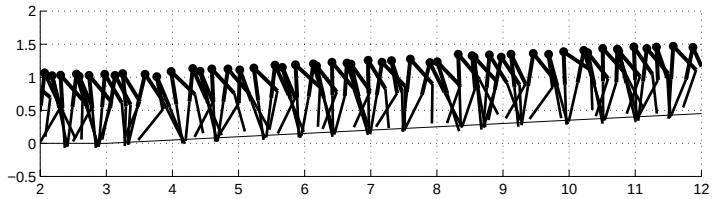


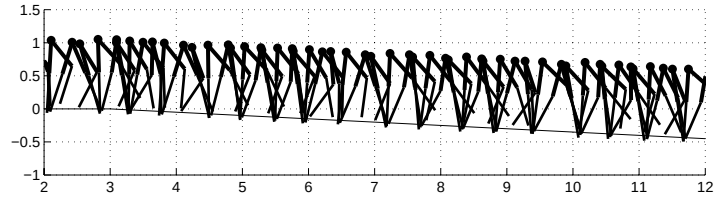
図 3.9 地面の変化と位相図



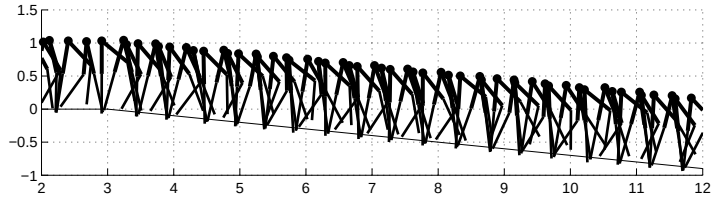
(a) 上り (急)



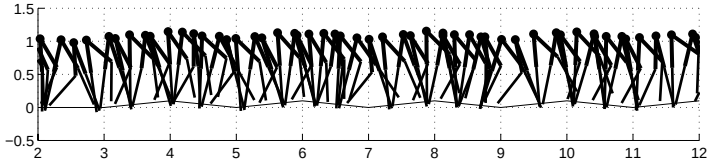
(b) 上り (緩)



(c) 下り (緩)

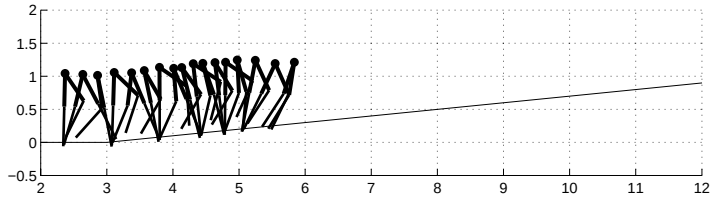


(d) 下り (急)

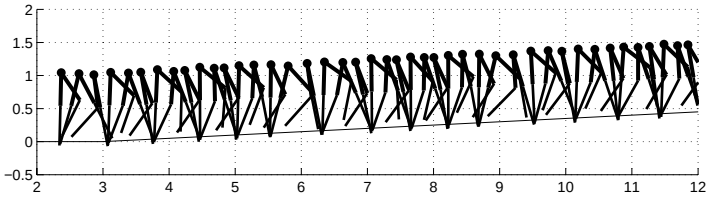


(e) デコボコ道

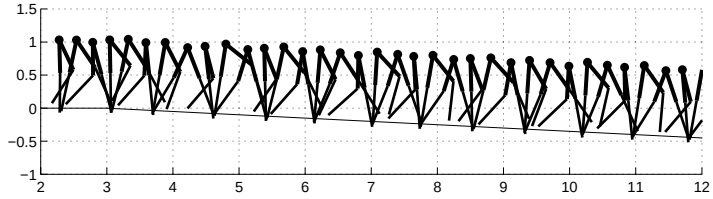
図 3.10 $\mathbf{a}_{RL}$ を用いた場合の歩容



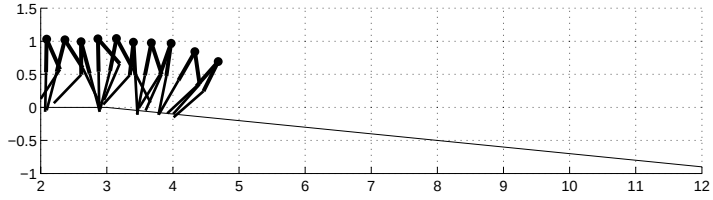
(a) 上り (急)



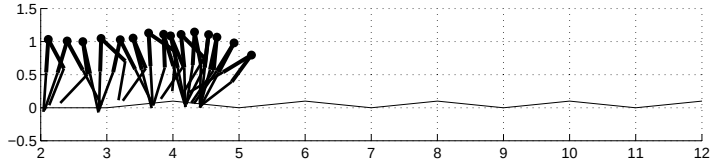
(b) 上り (緩)



(c) 下り (緩)



(d) 下り (急)



(e) デコボコ道

図 3.11 $\mathbf{a}_{HT}$ を用いた場合の歩容

第4章　方策勾配法に基づいた CPG-actor-critic モデル

3 章では, 価値関数に基づいて CPG-actor-critic モデルの学習を行う手法を提案した. 提案手法により安定な 2 足歩行を生成する CPG コントローラが得られたが, 学習過程は不安定であった. Konda ら [35] によって確率的方策勾配法 [36] [37] に基づいた actor-critic モデルが提案された. この手法は学習の収束が保証されている. 同様の結果は Sutton らによっても導かれている [38]. 本章では, CPG コントローラに対する学習を行う手法として, 確率的方策勾配法に基づいた CPG-actor-critic モデルを提案する.

4.1. 確率的方策勾配法

本節では, 確率的方策勾配法について説明する. 時刻 t において, actor が制御対象の状態 $\mathbf{s}(t)$ を観測し, 制御信号 $\mathbf{u}(t)$ を確率的方策 π に従い出力する. 制御対象は制御信号 $\mathbf{u}(t)$ とシステムのダイナミクスに従い $\mathbf{s}(t+1)$ に状態を遷移させる. この $\mathbf{s}(t)$ から $\mathbf{s}(t+1)$ への状態遷移確率を $p(\mathbf{s}(t+1)|\mathbf{s}(t), \mathbf{u}(t))$ と書く. また, 状態遷移と同時に同時に学習システムは即時報酬 $r(\mathbf{s}(t), \mathbf{u}(t))$ を与えられる. ここで, $\mathcal{S}$ と $\mathcal{U}$ はそれぞれ制御対象の状態空間と行動空間を示す.

学習エージェントの出力する制御信号は確率的方策

$$\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s}) = p(\mathbf{u}|\mathbf{s}) \tag{4.1}$$

に従うと仮定する. 方策とは被制御システムの状態が $\mathbf{s}$ の時に制御信号 $\mathbf{u}$ が出力される確率のことである. また, 確率的方策 $\pi_{\boldsymbol{\theta}}$ はパラメータ $\boldsymbol{\theta}$ によって特徴づけられる. また, $\pi_{\boldsymbol{\theta}}$ はそれぞれのパラメータ θ_i によって微分可能で, すなわち, $\frac{\partial}{\partial \theta_i}\pi_{\boldsymbol{\theta}}$ が存在すると仮定する. さらに $\pi_{\boldsymbol{\theta}}$ の下で, 被制御システムの状態変数はシステムの初期状態に依存しない定常分布 $D_{\boldsymbol{\theta}}(\mathbf{s})$ を持つと仮定する.

本章で定式化する強化学習法の目的は

$$\rho(\boldsymbol{\theta}) = \int_{\mathbf{s}\in\mathbf{S},\mathbf{u}\in\mathbf{U}} d\mathbf{s}d\mathbf{u}r(\mathbf{s},\mathbf{u})D_{\boldsymbol{\theta}}(\mathbf{s})\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s}) \tag{4.2}$$

で定義される 1 ステップあたりの期待報酬 $\rho(\boldsymbol{\theta})$, あるいは

$$\rho(\boldsymbol{\theta}) = \mathrm{E}\left[\sum_{t=1}^{\infty}\gamma^{t-1}r(\mathbf{s}(t),\mathbf{u}(t))\right] \tag{4.3}$$

で定義される累積期待報酬を最大化する方策パラメータ $\boldsymbol{\theta}$ を求めることである.

ここで, 1 ステップあたりの期待報酬の最大化について定式化を行う. 状態 $\mathbf{s}$ に対する状態価値関数は

$$V_{\boldsymbol{\theta}}(\mathbf{s}) = E\left[\sum_{t=0}^{\infty}\left(r(\mathbf{s}(t),\mathbf{u}(t)) - \rho(\boldsymbol{\theta})\right)|\mathbf{s}(0)=\mathbf{s}\right] \tag{4.4}$$

で定義される. $V_{\boldsymbol{\theta}}(\mathbf{s})$ は状態 $\mathbf{s}$ の良さを表していて, 1 ステップあたりの期待報酬 $\rho(\boldsymbol{\theta})$ と状態 $\mathbf{s}$ を初期状態とした場合の平均報酬との差を示している. この状態価値関数は 4.1 章の状態価値関数とは異なり, 相対価値と呼ばれるものである. この定義によれば, 状態価値関数 $V_{\boldsymbol{\theta}}(\mathbf{s})$ はポアソン方程式

$$\rho(\boldsymbol{\theta}) + V_{\boldsymbol{\theta}}(\mathbf{s}) = \int_u d\mathbf{u}\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})\left[r(\mathbf{s},\mathbf{u}) + \int_{\mathbf{s}'} d\mathbf{s}p(\mathbf{s}'|\mathbf{s},\mathbf{u})V_{\boldsymbol{\theta}}(\mathbf{s}')\right] \tag{4.5}$$

の解となっている. ここで $\mathbf{s}'$ は, 次時刻におけるシステムの状態である [39]. また, 行動価値関数を

$$Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) = r(\mathbf{s},\mathbf{u}) - \rho(\boldsymbol{\theta}) + \int_{\mathbf{s}'} d\mathbf{s}'p(\mathbf{s}'|\mathbf{s},\mathbf{u})V_{\boldsymbol{\theta}}(\mathbf{s}') \tag{4.6}$$

と定義する. 1 ステップあたり期待報酬 $\rho(\boldsymbol{\theta})$ の方策パラメータの i 番目の要素 θ_i に関する勾配は

$$\nabla_i\rho(\boldsymbol{\theta}) = \int_{\mathbf{s},\mathbf{u}} d\mathbf{s}d\mathbf{u}D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})\psi_i(\mathbf{s},\mathbf{u}) \tag{4.7}$$

と計算できる (導出は付録 C). ここで,

$$\psi_i(\mathbf{s},\mathbf{u}) = \nabla_i \ln\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s}) \tag{4.8}$$

で, ∇_i は方策パラメータ $\boldsymbol{\theta}$ の i 番目の要素 θ_i に関する偏微分, すなわち $\partial/\partial\theta_i$ である [35] [39] [38].

式 (4.7) の方策の勾配を計算するためには, 行動価値関数 $Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})$ を推定する必要がある. ここで, $\mathcal{S}\times\mathcal{U}$ 上の二つの実数値関数 Q_1 と Q_2 の内積 $\langle\cdot,\cdot\rangle_{\boldsymbol{\theta}}$ を

$$\langle Q_1(\mathbf{s},\mathbf{u}),Q_1(\mathbf{s},\mathbf{u})\rangle_{\boldsymbol{\theta}}=\int_{\mathbf{s},\mathbf{u}}d\mathbf{s}d\mathbf{u}D_\theta(\mathbf{s},\mathbf{u})Q_1(\mathbf{s},\mathbf{u})Q_2(\mathbf{s},\mathbf{u}) \tag{4.9}$$

と定義する. 式 (4.7) の方策勾配 $\nabla_i\rho(\boldsymbol{\theta})$ は行動価値関数 $Q_\theta(\mathbf{s},\mathbf{u})$ と基底関数 $\psi_i(\mathbf{s},\mathbf{u})$ の内積で表すことができる. よって, 真の行動価値関数 $Q_\theta(\mathbf{s},\mathbf{u})$ は $\mathcal{S}\times\mathcal{U}\to R$ の写像であるが, 方策勾配を求めるためには行動価値関数と基底関数 $\psi_i(\mathbf{s},\mathbf{u})$ の内積の定常分布の下での期待値

$$\nabla_i\rho(\boldsymbol{\theta})=\langle Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}),\psi_i(\mathbf{s},\mathbf{u})\rangle \tag{4.10}$$

を計算すれば十分である. よって, 真の行動価値関数そのものを学習する必要はなく, Q 関数の基底関数 Ψ によって張られる空間への写像 $\hat{Q}_{\boldsymbol{\theta}}$ を学習すれば良い. 多くの場合, Ψ によって張られる空間は元の行動価値関数の入力空間 $\mathcal{S}\times\mathcal{U}$ よりも低次元であるため, 写像 $\hat{Q}_{\boldsymbol{\theta}}$ に対する学習は真の行動価値関数に対する学習に比べて容易である [35].

Actor-critic モデル

次に, Konda ら [35] が導出した確率的勾配法に基づいた actor-critic モデルについて説明する. Critic は Q 関数の基底 $\{\phi_j,j=1,\dots,m\}$ で張られる空間への写像 $\hat{Q}_{\boldsymbol{\theta}}$ を近似する. ただし, 基底 Φ は $\{\psi_i,i=1,\dots,n\}$ を含むものとする. また, critic は以下に示すような線形近似アーキテクチャで表現されるとする.

$$Q^w_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})=\sum_{j=1}^m w_j\phi_j(\mathbf{s},\mathbf{u}) \tag{4.11}$$

ここで, $\mathbf{w}=\{w_j\}$ は critic のパラメータベクトルである.

Critic のパラメータは TD(λ) 法に基づいて更新される ($\lambda\in(0,1)$). j 番目の基底関数 ϕ_j に対する適格度トレース Z_j は

$$Z_j(t)=\lambda Z_j(t-1)+\phi_j(\mathbf{s}(t),\mathbf{u}(t)) \tag{4.12}$$

のように計算される. ここで適格度トレースは $Z_j(t)=\sum_{\tau=0}^t\lambda^{t-\tau}\phi_j(\mathbf{s}(\tau),\mathbf{u})$ で計算される基底関数 ϕ_j の報酬に対する寄与度を示したものである. これを用いると, 1 ステップあたりの期待平均報酬 $\rho(\boldsymbol{\theta})$ と critic のパラメータは

$$\rho(\boldsymbol{\theta}):=\rho(\boldsymbol{\theta})+\eta_t\left(r(\mathbf{s}(t),\mathbf{u}(t))-\rho(\boldsymbol{\theta})\right) \tag{4.13}$$

$$\mathbf{w} := \mathbf{w} + \eta_t \delta(t) \mathbf{Z}(t) \tag{4.14}$$

のように更新される [40]. ここで $\delta(t)$ は TD 誤差であり,

$$\delta(t) = r(\mathbf{s}(t), \mathbf{u}(t)) - \rho(\boldsymbol{\theta}) + \sum w_j \phi_j(\mathbf{s}(t+1), \mathbf{u}(t+1)) - \sum w_j \phi_j(\mathbf{s}(t), \mathbf{u}(t)) \tag{4.15}$$

で計算される.

Actor は確率的方策 $\pi_{\boldsymbol{\theta}}$ に従って, 制御信号 $\mathbf{u}$ を出力する制御器で, actor のパラメータは 1 ステップあたりの期待平均報酬 $\rho(\boldsymbol{\theta})$ を増加させるように勾配法によって

$$\theta_i := \theta_i + \beta(t) \Gamma(\mathbf{w}) \left[\sum_j r_j \phi_j(\mathbf{s}(t+1), \mathbf{u}(t+1)) \right] \psi_i(\mathbf{s}(t+1), \mathbf{u}(t+1)) \tag{4.16}$$

のように更新される. ここで, $\beta(t)$ は学習係数で, $\Gamma(\mathbf{w})$ は現在の critic のパラメータに依存して actor の学習速度を調整するための項である.

この手続きによって, 学習が収束するためには学習係数に対する以下のような条件が必要である [35].

- 学習係数 β と γ は決定論的に減少する系列で, $\sum_k \beta_k = \sum_k \gamma_k = \infty$, $\sum_k \beta_k^2 < \infty$, $\sum_k \gamma_k^2 < \infty$ と $\sum_k (\beta_k / \gamma_k)^d < \infty$ を満たす必要がある. β_k と γ_k は k 回目のパラメータ更新における学習係数を示す.

- 関数 $\Gamma(\mathbf{w})$ は $|\mathbf{w}|\Gamma(\mathbf{w}) \in [C_1, C_2], \forall \mathbf{w} \in \mathcal{R}^m$ とある正数 $C_1 < C_2$ に対して, $|\Gamma(\mathbf{w}) - \Gamma(\hat{\mathbf{w}})| \leq C_2 |\mathbf{w} - \hat{\mathbf{w}}| / (1 + |\mathbf{w}| + |\hat{\mathbf{w}}|), \forall \mathbf{w}, \hat{\mathbf{w}} \in \mathcal{R}^m$ を満たす.

4.2. 方策勾配法に基づいた CPG-actor-critic モデル

CPG コントローラに対して確率的方策勾配法を適用する場合には幾つかの問題がある. この方法では, CPG コントローラが actor として扱われるが, CPG コントローラの出力は CPG ニューロンの状態に依存する. 制御対象の物理システムの状態が同一であっても CPG コントローラの出力が異なるものとなり, すなわち, CPG コントローラによる確率的方策は非定常な方策となる. 多くの強化学習アルゴリズムは定常方策の下で適用されてきた. 特に, 確率的方策勾配法においては方策が定常であり, ある方策の下で制御対象の状態変数が定常分布を持つという仮定を行う. また, actor は定常分布の下での平均報酬, または期待累積報酬を最大化するように学習を行うように設計されている.

さらに, 通常の強化学習法では時間的に局所的な誤差 (TD 誤差) に基づいてオンライン的に学習を行うが, CPG コントローラが再帰結合型ニューラルネットワークで実装されるため, 再帰結合型ニューラルネットワークの学習を行うためには例えば時間逆向きバックプロパゲーション [28] 等を用いる必要があり, これらのアルゴリズムは過去の履歴を保持する必要があるためにオンライン学習には適していない事や計算量が大きい等の問題がある.

よって, CPG コントローラの学習に対して確率的方策勾配法を適用する場合も 3 章と同様に, 物理システムと神経振動子ネットワーク内のニューロンを一つの動的システム (CPG 結合システム) として扱い, CPG コントローラによって物理システムを制御するタスクから, 相互結合を持たない actor によって CPG 結合システムを制御するモデルへと変更することが有用である.

この場合, actor の出力は制御対象である CPG 結合システムの状態にだけ依存するため, 確率的方策は定常である. よって, 以下に示すように actor のパラメータを確率的方策勾配法を用いて学習することができる.

4.2.1 方策勾配法に基づいた学習法

CPG-actor-critic モデルに方策勾配法を適用する場合, actor は CPG ニューロンの状態変数 $\boldsymbol{\nu}$ と物理システムの状態変数 $\mathbf{x}$, すなわち CPG 結合システムの状態 $\mathbf{s} = \{\boldsymbol{\nu}', \mathbf{x}'\}'$ を観測し, 確率的方策 $\pi_{\boldsymbol{\theta}}$ に従い間接制御信号 $\mathbf{u}$ を出力する. ここで $(')$ はベクトル転置を示す. Actor が CPG 結合システムを制御するモデルと見なすことにより 4.1 節で示したように方策パラメータ $\boldsymbol{\theta}$ の学習を行うことができる.

エピソード学習

時刻 t において, actor は CPG 結合システムの状態 $\mathbf{s}(t)$ を観測し, 方策 $\pi_{\boldsymbol{\theta}}$ に従って間接制御信号 $\mathbf{u}(t)$ を出力する.

この後, CPG 結合システムは式 (2.1)〜(3.6) に従い状態を $(\mathbf{s}(t+1))$ に変化させる.

同時に, critic は即時報酬 $r(\mathbf{s}(t), \mathbf{u}(t))$ を観測し, 式 (4.14) と式 (4.13) に従ってそれぞれ critic のパラメータ $\mathbf{r}$ と期待平均報酬の推定値 $\rho(\boldsymbol{\theta})$ を更新する.

学習システムは一定時間 (t_{max}) が経過するか, 物理システムが終了状態になるまで, 上記の actor による制御と critic の学習を繰り返す. すなわち, この間は actor のパラ

メータを固定している．また, 制御を行っている間の CPG 結合システムの状態の系列 $\{\mathbf{s}(t), \mathbf{u}(t) | t = 0, 1, \ldots, t_{max}\}$ は次ステップの actor の学習のために記録される．

次に, 記録された各時刻の状態 $(\boldsymbol{\nu}(t), \mathbf{x}(t)), t = 0, 1, \ldots, t_{max}$ に対して行動評価関数の値を増加させるように actor のパラメータを更新する．期待平均報酬 $\rho(\boldsymbol{\theta})$ を増大させるように, 式 (4.16) に従い actor のパラメータ $\boldsymbol{\theta}$ を更新する [38] [35]．このパラメータ更新は疑似オンライン的に行われる．

以上の 2 つの学習ステップを 1 エピソードとし, 強化学習はこのエピソードを繰り返すことで行われる．本手法では actor と critic に対して同時にオンライン学習を行うことは可能である．しかし, actor の学習によって critic の学習目標が変化し, critic の学習によって actor の学習目標が変化するという相互に依存した関係があるため, actor 学習時には critic を固定し, critic 学習時には actor を固定することで学習が安定する．本手法ではこの安定性を実現する一つの方法としてエピソードごとに critic の学習と actor の学習を交互に行う方法を用いる．

4.3. 実験:方策勾配法

CPG-actor-critic モデルに基づく強化学習法を 2 足歩行ロボットシミュレータ [10] に適用した．シミュレーション実験の目的は, ロボットが安定した歩行運動を行うような CPG コントローラのパラメータを自律的に獲得することである．

実験条件

CPG 結合システムのダイナミクスは時間間隔 0.0001 秒の Runge-Kutta 法を用いた積分によって計算した．また, 学習システムは 0.01 秒毎にシステムの状態を観測し, 制御信号を出力した．また, 各リンクの角速度の観測値はサンプリング時刻における値ではなく, 0.01 秒間の平均を用いた．

基本 CPG として文献 [10] で用いられた神経振動子ネットワークを用いた．また, 今回の実験では, 学習を容易にするために, actor のパラメータの内, 学習できるパラメータを制限した．ニューロン間の相互結合の重みパラメータを固定, すなわち $W_{ij}^{act} \equiv 0$ とした．

また, CPG 結合システムの状態が $\mathbf{s}$ であるときに間接制御信号 $\mathbf{u}$ が出力される確率を

$$\pi_\theta(\mathbf{u}|\mathbf{s}) = \mathcal{N}(\bar{\mathbf{u}}, \theta_9) = (2\pi)^{-1/2}\theta_9^{-1} \exp\left\{-\frac{(\mathbf{u}-\bar{\mathbf{u}})'(\mathbf{u}-\bar{\mathbf{u}})}{2\theta_9^2}\right\} \tag{4.17}$$

とした. ここで $\bar{\mathbf{u}}$ は

$$\begin{aligned}
\bar{u}_1 &= \theta_1 X_1 - \theta_2 X_2 + \theta_3 X_3 + \theta_4 X_6 \\
\bar{u}_2 &= \theta_1 X_2 - \theta_2 X_1 + \theta_3 X_4 + \theta_4 X_5 \\
\bar{u}_3 &= \theta_5 X_4 \\
\bar{u}_4 &= \theta_5 X_3 \\
\bar{u}_5 &= -\theta_6 X_3 - \theta_7 X_4 - \theta_8 X_7 \\
\bar{u}_6 &= -\theta_6 X_4 - \theta_7 X_3 - \theta_8 X_8
\end{aligned} \tag{4.18}$$

とした. ここで $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_9\}$ は actor のパラメータベクトルである. また, $2i-1$ 番目と $2i$ 番目の CPG ニューロンに対する外部入力はそれぞれ $I_{2i-1} = u_i$ と $I_{2i} = -u_i$ とする.

Critic の基底関数は actor のパラメータ θ_i によって決まる $\phi_i = \psi_i = \nabla \ln \pi_\theta(\mathbf{u}|\mathbf{s}), i = 1, \ldots, 8$ だけを用いた.

直接報酬 $r(\boldsymbol{\nu}(t), \mathbf{x}(t), \mathbf{u}(t))$ は次時刻のロボットの状態 $\mathbf{x}(t+1)$ だけに依存するものとして, $\tilde{r}(\mathbf{x}(t+1))$ で与え, 3 章で与えた報酬と同様に

$$\begin{aligned}
\tilde{r}(\mathbf{x}) &= k_h r_h(\mathbf{x}) + k_s r_s(\mathbf{x}) \\
r_h(\mathbf{x}) &= h_1 - \min(h_4, h_5) - H \\
r_s(\mathbf{x}) &= \begin{cases} x_7 & \text{if } |x_7| < 1 \\ x_7/|x_7| & \text{otherwise} \end{cases}
\end{aligned} \tag{4.19}$$

と定義した. ここで, h_i $(i = 4, 5)$ は link-i のかかとの高さで, x_7 は link-1 の水平方向に対する速度を示している. 報酬 $r_h(\mathbf{x})$ は腰の位置が高い程大きくなり, ロボットが転倒しないように学習するのに役立つ. 報酬 $r_s(\mathbf{x})$ は右方向への速度が大きいほど大きくなる. この報酬項は, ロボットが前方向に歩くのを促進する働きがある. k_h と k_s はそれぞれ r_h と r_s に対する重みで, H はしきい値である.

1 エピソードは最大 5 秒間とし, 5 秒以内にロボットが転倒した場合はその時点でエピソードを終了させる. 各エピソードのロボットの初期状態は脚をわずかに開いて静止した状態, ニューロンの初期状態は全てのニューロンの内部状態を 0 とした. また, 初期状態の脚の角度はエピソード毎に一定の角度内でランダムに異なる角度を選んだ. 強化学習はこのような学習手続きを繰り返すことによって行われた. また, actor の学習係数を

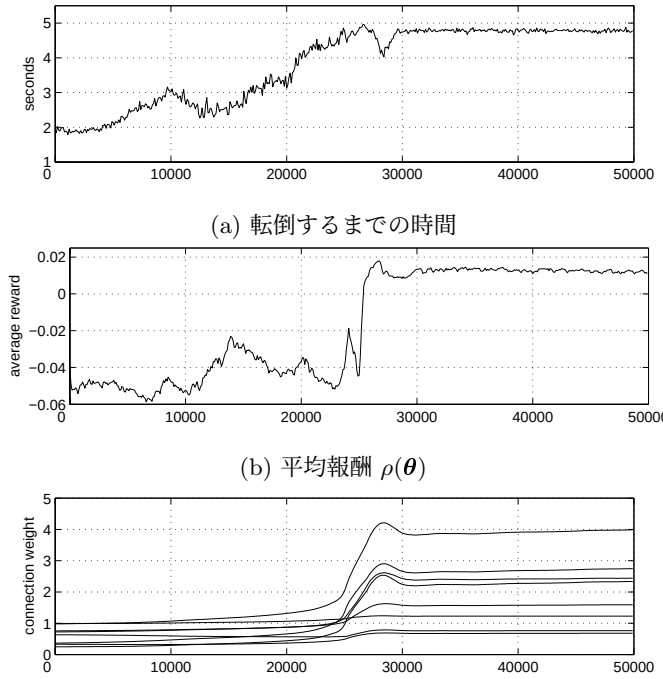
制御する関数 $\Gamma(\mathbf{w})$ は

$$\Gamma(\mathbf{w}) = \frac{1}{1 + |\mathbf{w}|} \tag{4.20}$$

とした. ここで, $|\cdot|$ はベクトルのノルムを示す. この関数は前述の収束のための条件を満たす.

4.3.1 歩行運動の獲得

本実験では, 提案手法により 2 足歩行運動を可能にする CPG コントローラが学習によって得られるかを確かめる. Actor パラメータ $\boldsymbol{\theta}$ の初期値としてベクトルの各成分を小さな乱数で与えた. このパラメータでは, 図 4.2(a) に示すように 2 足歩行ロボットはすぐに転倒する. また, 式 (4.19) の重み k_h と k_s をそれぞれ 0.1 と 0.002 とし, しきい値 H を 0.9 とした. ロボットが転倒した場合に大きな罰を与えるために, 式 (4.19) に r_h が 0 以下となった場合に $-r_h^2$ という負の報酬を追加した.

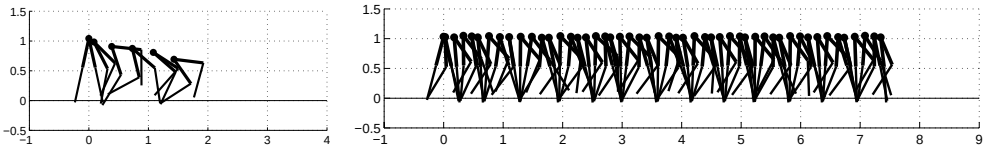


(a) 転倒するまでの時間

(b) 平均報酬 $\rho(\boldsymbol{\theta})$

(c) Actor のパラメータの変化 $\{\theta_1, \ldots, \theta_8\}$

図 4.1 学習過程



(a) 学習前の歩容　　(b) 50000 学習エピソード後の歩容

図 4.2 学習前と学習後における歩容

図 4.1 に学習曲線を示す. 横軸は学習エピソード数を示し, 縦軸は期待平均報酬 $\rho(\boldsymbol{\theta})$ を示す. 図 4.1(a) にロボットが転倒するか, 5.0 秒間経過することでエピソードが終了するまでの秒数について示す. 図 4.1(b) に 1 タイムステップあたりの平均報酬を示す. また, これらの図は 100 エピソード分の移動平均である. また, この時の方策パラメータの変化を図 4.1(c) に示す. 約 25000 回の学習エピソード後に, ロボットは 5 秒間転倒せずに歩行ができるようになり, 期待平均報酬が大きくなる. 図 4.2(a) と 4.2(b) にそれぞれ学習前と 50000 回の学習エピソード後の歩容を示す.

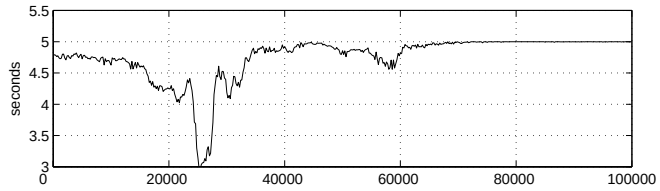
追加学習

ここで得られた actor のパラメータを初期値とし, 再度, 提案手法を用いて学習を行った. 図 4.3 に学習曲線を示す. 約 25000 学習エピソード後に一度大きく報酬が落ち込んでいるが, その後, 平均報酬は落ち込む以前よりも大きな値になっている. 追加学習により, ほぼ転倒しない CPG コントローラが獲得された. 学習後の結合重みパラメータは $\boldsymbol{\theta}_{pg1} = \{0.72, 0.73, 4.52, 1.25, 0.71, 3.99, 5.57, 2.42\}$ であった.

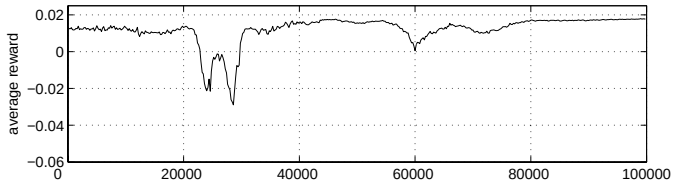
4.3.2 パラメータの調整

この実験では提案手法により actor のパラメータが収束するかを調べた. actor のパラメータ θ_i の初期値を文献 [10] で用いられた値 $\boldsymbol{\theta}_{HT} = \{1.50, 1.00, 1.50, 1.50, 3.00, 1.50, 3.00, 1.50\}$ とし, 学習を行った. また, 式 (4.19) の重み k_h と k_s はそれぞれ 0.1 と 0.002 とし, しきい値 H は 0.8 とした. Actor のパラメータが $\boldsymbol{\theta}_{HT}$ の時, 2 足歩行ロボットは転倒せずに安定した歩行を行うが, 歩き出しの段階では転倒する場合がある.

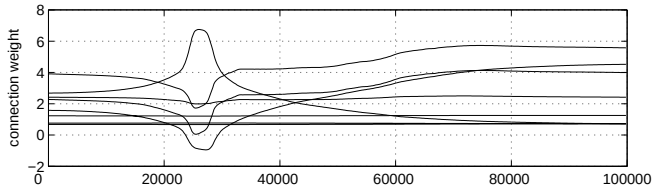
図 4.4(a) と 図 4.4(b) はそれぞれエピソードが終了するまでの時間と期待平均報酬 $\rho(\boldsymbol{\theta})$ を示している. 学習初期に期待平均報酬が下がるがその後, 学習の進行に従ってが増大し, 同時にロボットが転倒する割合が減少していることが分かる. 50000 学習エピ



(a) 転倒するまでの時間



(b) 平均報酬 $\rho(\boldsymbol{\theta})$



(c) Actor のパラメータの変化 $\{\theta_1, \ldots, \theta_8\}$

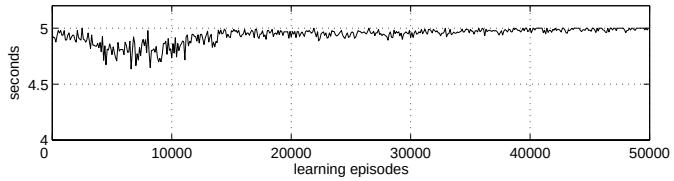
図 4.3 追加学習における学習過程

ソード後, actor のパラメータは $\boldsymbol{\theta}_{pg2} = \{1.46, 0.96, 1.39, 1.49, 2.73, 1.32, 2.72, 1.23\}$ となった.

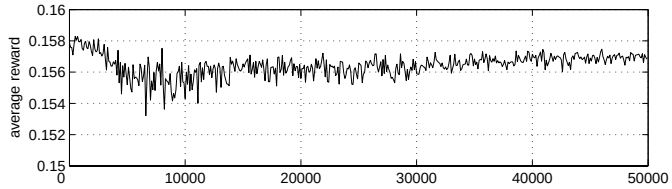
4.3.3 実験結果

提案手法を 2 足歩行運動の自律的な学習課題に適用し, すぐに転倒するようなパラメータから学習を始めた場合にはより長時間転倒せずに歩行を行う事ができ, 安定した歩行が可能なパラメータから学習を始めた場合にはより安定した歩行が可能な CPG コントローラが得られた. さらに, 学習後の actor のパラメータを初期値としてさらに学習を行うことで, より安定な歩行が可能となる CPG コントローラが得られた. 追加学習によってより良い制御器が得られる理由としては学習初期においては学習係数が比較的大きくより探索しやすいが, 学習が進行に従い学習係数が小さくなることから探索範囲が狭くなる傾向があることが考えられる.

学習過程は安定しており, 学習の初期パラメータの付近で大きな報酬を得られる局所最



(a) 転倒するまでの時間



(b) 平均報酬

図 4.4 学習曲線

適なパラメータを学習していると考えられる. よって, 大域的に最適なパラメータを獲得するためには何らかの探索手法が必要であると考えられる. また, 学習速度が遅いことも本手法の問題点であり, 学習を高速にする必要がある.

4.4. 自然方策勾配法に基づいた学習法

方策勾配法に基づいた CPG-actor-critic モデルの問題点として学習速度が遅いことがある. Kakade [41] によって高速な学習アルゴリズムである自然勾配法 [42] を用いた自然方策学習法 (Natural Policy Gradient Method) が提案された. この節では自然方策勾配法を CPG-actor-critic モデルに適用する.

本節では式 (4.3) で与えられる累積期待報酬を最大化する方策パラメータ $\boldsymbol{\theta}$ を求める. ここで, 状態 $\mathbf{s}$ に対する状態価値関数を

$$V_{\boldsymbol{\theta}}(\mathbf{s}) \equiv E\left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}(t), \mathbf{u}(t)) | \mathbf{s}(0) = \mathbf{s}, \pi_{\boldsymbol{\theta}}\right] \tag{4.21}$$

のように定義する. ここで $\gamma \in (0, 1]$ は割引率である. また, Q 関数とも呼ばれる行動価値関数は

$$Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{u}) = r(\mathbf{s}, \mathbf{u}) + \gamma \int_{\mathbf{s}'} d\mathbf{s}' p(\mathbf{s}' | \mathbf{s}, \mathbf{u}) V_{\boldsymbol{\theta}}(\mathbf{s}') \tag{4.22}$$

と定義される. ここで $\mathbf{s}'$ は次時刻におけるシステムの状態である. Q 関数は現在の状態と行動を行い, その後現在の方策 $\pi_{\boldsymbol{\theta}}$ に従って行動決定した場合に期待累積報酬である.

期待累積報酬 (4.22) の方策パラメータ θ_i に関する偏微分は 4.1 節と同様に

$$\frac{\partial \rho(\boldsymbol{\theta})}{\partial \theta_i} = \int_{\mathbf{s},\mathbf{u}} d\mathbf{s} d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \psi_i(\mathbf{s},\mathbf{u}) Q_{\theta}(\mathbf{s},\mathbf{u})$$

となる. ここで ψ_i は $\ln \pi_{\boldsymbol{\theta}}$ の方策パラメータ θ_i に関する微分, すなわち $\partial \ln \pi_{\boldsymbol{\theta}} / \partial \theta_i$ である [38] [39] [35].

この方策勾配 (4.7) を推定するために, Q 関数に対する関数近似器として 4.1 と同様の線形近似アーキテクチャ, すなわち

$$f^w(\mathbf{s},\mathbf{u}) = \sum_{i=1}^{n} \psi_i(\mathbf{s},\mathbf{u}) w_i \tag{4.23}$$

を用いる. ここで $\mathbf{w}$ はパラメータである. $f^w(\mathbf{s},\mathbf{u})$ のパラメータが

$$\mathbf{w} = \arg\min_{\mathbf{w}} \left(Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) - f^w(\mathbf{s},\mathbf{u}) \right)^2$$

を満たすとき, 真の Q 関数 $Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})$ の代りに $f^w(\mathbf{s},\mathbf{u})$ を用いて計算した方策勾配 (4.7) には偏りが無いことが示された [38]. また, f^w は $0 = \int_{\mathbf{u}} d\mathbf{u} \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s}) f^w(\mathbf{s},\mathbf{u}), \forall \mathbf{s} \in \mathcal{S}$ を満たすので, 状態と行動の効用を示す f^w は行動評価関数よりは, $A^{\pi}(\mathbf{s},\mathbf{u}) = Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) - V^{\pi}(\mathbf{s})$ で定義され, 状態 $\mathbf{s}$ の元での行動 $\mathbf{u}$ の良さを示す advantage function に近い性質を持つ. さらに, $\int_{\mathbf{u}} d\mathbf{u} \psi_i(\mathbf{s},\mathbf{u}) = 0, \forall \mathbf{s} \in \mathcal{S}$ であるため, 方策勾配 (4.7) は任意の $\mathbf{s}$ の関数 $\hat{V}(\mathbf{s})$ が f^w に追加されても推定値にバイアスが加わらない:

$$\frac{\partial \rho(\boldsymbol{\theta})}{\partial \theta_i} = \int_{\mathbf{s},\mathbf{u}} d\mathbf{s} d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \psi_i(\mathbf{s},\mathbf{u}) \left(f^w(\mathbf{s},\mathbf{u}) + \hat{V}(\mathbf{s}) \right), \tag{4.24}$$

[38] [35] [43].

4.4.1 自然方策勾配法

価値関数が式 (4.23) のような線形近似アーキテクチャで近似される場合, 方策勾配 (4.7) の代りに自然方策勾配を用いて方策パラメータ $\boldsymbol{\theta}$ の更新を行うことができる. [41] [43].

価値の学習の目的は真の Q 関数と関数近似器 (4.23) の差を最小にすることである. 提案手法では二乗誤差

$$e(\mathbf{w}) = \int_{\mathbf{s},\mathbf{u}} d\mathbf{s} d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \left(f^w(\mathbf{s},\mathbf{u}) - Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \right)^2 \tag{4.25}$$

を最小にするパラメータを $\tilde{\mathbf{w}}$ とすると $\frac{\partial e(\tilde{\mathbf{w}})}{\partial w_i} = 0$ を満たす. この時

$$\int_{\mathbf{s},\mathbf{u}} d\mathbf{s}d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})\boldsymbol{\psi}(\mathbf{s},\mathbf{u})\left(\boldsymbol{\psi}(\mathbf{s},\mathbf{u})'\tilde{\mathbf{w}} - Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})\right) = 0$$

より,

$$\int_{\mathbf{s},\mathbf{u}} d\mathbf{s}d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})\boldsymbol{\psi}(\mathbf{s},\mathbf{u})\boldsymbol{\psi}(\mathbf{s},\mathbf{u})'\tilde{\mathbf{w}} = \int_{\mathbf{s},\mathbf{u}} d\mathbf{s}d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})\boldsymbol{\psi}(\mathbf{s},\mathbf{u})Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \tag{4.26}$$

となる. また, 方策 $\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})$ の定常分布 $D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})$ の下での Fisher 情報量行列 $F(\boldsymbol{\theta})$ の平均は

$$\begin{aligned} F(\boldsymbol{\theta}) &= \int_{\mathbf{s},\mathbf{u}} d\mathbf{s}d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})\left[\frac{\partial \log \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial \theta_i}\frac{\partial \log \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial \theta_j}\right] \\ &= \int_{\mathbf{s},\mathbf{u}} d\mathbf{s}d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})\boldsymbol{\psi}(\mathbf{s},\mathbf{u})\boldsymbol{\psi}(\mathbf{s},\mathbf{u})' \end{aligned} \tag{4.27}$$

と計算できる. よって, 式 (4.26) の左辺は $F(\boldsymbol{\theta})\tilde{\mathbf{w}}$ となり, また, 右辺は方策勾配 (4.7) である. よって, 式 (4.26) は $F(\boldsymbol{\theta})\tilde{\mathbf{w}} = \nabla\rho(\boldsymbol{\theta})$ となり,

$$\tilde{\mathbf{w}} = F(\boldsymbol{\theta})^{-1}\nabla\rho(\boldsymbol{\theta}) \tag{4.28}$$

が自然方策勾配であることが導かれる.

4.4.2 学習アルゴリズム

提案手法は 4.1 節と同様に actor-critic モデルに基づくものである. Actor は確率的方策 $\pi_{\boldsymbol{\theta}}$ に従って制御信号を出力し, また, critic は Q 関数を近似を行うもので, $\{\psi_{\theta_i}|i = 1,\ldots,n\}$ を含む基底関数 $\{\phi_j|j = 1,\ldots,m\}$ の線形和で表されるものとする:

$$Q^w_{\boldsymbol{\theta}} = \sum_{j=1}^{m} w_j\phi_j(\mathbf{s},\mathbf{u}) \tag{4.29}$$

ここで, $\mathbf{w} = \{w_j\}$ は critic のパラメータベクトルである. また, 基底関数 $\phi_i = \psi_i, i = 1,\ldots,n$ は actor のパラメータによって自動的に決定され, それ以外の基底関数 $\phi_i, i = n+1,\ldots,m$ は状態 $\mathbf{s}$ を入力とする任意の関数とする. 後者の基底関数は上述の状態価値関数 $V(\mathbf{s})$ に対応するもので, これらの基底を加えることによって方策勾配 (4.7) の推定値の分散が減少することが期待できる.

また, 式 (4.25) を最小にするパラメータ $\tilde{\mathbf{w}}$ が得られると, actor のパラメータ $\boldsymbol{\theta}$ は式 (4.28) に従い,

$$\theta_i := \theta + \eta_a \tilde{w}_i \tag{4.30}$$

と更新できる. ここで, η_a は actor の学習係数である.

LSQ

式 (4.25) を最小化するパラメータ $\tilde{\mathbf{w}}$ を求めるために最小二乗法に基づいた LSQ 法を用いる [44] [45] [46] [43]. 行動価値関数は固定された方策 $\pi_{\boldsymbol{\theta}}$ に対して自己無撞着等式

$$Q^w_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{u}) = r(\mathbf{s}, \mathbf{u}) + \gamma Q^w_{\boldsymbol{\theta}}(\mathbf{s}', \mathbf{u}') \tag{4.31}$$

を満たす必要がある. ここで, $\mathbf{s}'$ と $\mathbf{u}'$ はそれぞれ次時刻の状態と行動である.

On-line LSQ critic のパラメータ $\mathbf{w}$ の最小二乗推定値は以下の二乗誤差を最小化するものとする.

$$e(\mathbf{w}) = \int_{\mathbf{s},\mathbf{u}} D_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{u}) \left[r(\mathbf{s}, \mathbf{u}) + \left(\gamma \bar{\boldsymbol{\phi}}(\mathbf{s}', \mathbf{u}') - \boldsymbol{\phi}(\mathbf{s}, \mathbf{u}) \right)' \mathbf{w} \right]^2 \tag{4.32}$$

ただし,

$$\bar{\boldsymbol{\phi}}(\mathbf{s}', \mathbf{u}' | \mathbf{s}, \mathbf{u}) = \int_{\mathbf{s}',\mathbf{u}'} d\mathbf{s}' d\mathbf{u}' \boldsymbol{\phi}(\mathbf{s}', \mathbf{u}') p(\mathbf{u}'|\mathbf{s}') p(\mathbf{s}'|\mathbf{s}, \mathbf{u}) \tag{4.33}$$

である. データサンプル $\{\mathbf{s}(t), \mathbf{u}(t) | t = 0, 1, \ldots, T\}$ が与えられた場合, 定常分布は経験分布:

$$D_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{u}) = \sum_{t=0}^{T} \delta(\mathbf{s}(t), \mathbf{u}(t)) \tag{4.34}$$

で近似できる. ここで $\delta(\cdot)$ は Dirac のデルタ関数である. 古い方策による効果を忘却するために, 経験分布 (4.34) の計算に重み付き平均を導入する:

$$D_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{u}) = \alpha(T) \sum_{t=0}^{T} \left(\prod_{\tau=t+1}^{T} \beta(\tau) \right) \delta(\mathbf{s}(t), \mathbf{u}(t)). \tag{4.35}$$

ここで $\beta \in (0,1]$ は discount factor で, $\alpha(T) = \left(\sum_{t=1}^{T}\left(\prod_{\tau=t+1}^{T}\beta(\tau)\right)\right)^{-1}$ は正規化項である. また, 状態遷移確率をサンプルデータを用いて

$$p(\mathbf{s}'|\mathbf{s}(t),\mathbf{u}(t)) = \delta(\mathbf{s}(t+1))|\mathbf{s}(t),\mathbf{u}(t)). \tag{4.36}$$

のように近似すると, 式 (4.33) は

$$\bar{\boldsymbol{\phi}}(\mathbf{s}',\mathbf{u}'|\mathbf{s}(t),\mathbf{u}(t)) = \int_{\mathbf{s}',\mathbf{u}'} d\mathbf{s}'d\mathbf{u}'\boldsymbol{\phi}(\mathbf{s}',\mathbf{u}')p(\mathbf{u}'|\mathbf{s}')p(\mathbf{s}'|\mathbf{s}(t),\mathbf{u}(t)) \tag{4.37}$$

$$= \int_{\mathbf{u}'} d\mathbf{u}'\boldsymbol{\phi}(\mathbf{s}(t+1),\mathbf{u}')\pi_{\boldsymbol{\theta}}(\mathbf{u}'|\mathbf{s}(t+1)) \tag{4.38}$$

となる. この近似を行うと, システムの状態遷移が確率的である場合には推定値にバイアスが加わる. しかし, 決定論的に遷移するシステムではバイアスは加わらない [44]. さらに, $\int_{\mathbf{u}} d\mathbf{u}\psi_i(\mathbf{s},\mathbf{u})\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s}) = 0$ であり, 基底関数 $\phi_i(\cdot), i > n$ は行動 $\mathbf{u}$ を変数として持たないので,

$$\bar{\phi}_i(\mathbf{s}',\mathbf{u}'|\mathbf{s}(t),\mathbf{u}(t)) = \begin{cases} 0 & \text{for } i = 1,\ldots,n \\ \phi_i(\mathbf{s}(t+1)) & \text{for } i = n+1,\ldots,m \end{cases} \tag{4.39}$$

となる. よって, 二乗誤差 (4.32) は

$$e(\mathbf{w}) = \alpha(T)\sum_{t=0}^{T}\left(\prod_{\tau=t+1}^{T}\beta(\tau)\right)\left[r(\mathbf{s}(t),\mathbf{u}(t)) - \boldsymbol{\varphi}'(t)\mathbf{w}\right]^2 \tag{4.40}$$

と計算できる. ただし,

$$\boldsymbol{\varphi}(t) = \gamma\bar{\boldsymbol{\phi}}(\mathbf{s}(t+1),\mathbf{u}(t+1)) - \boldsymbol{\phi}(\mathbf{s}(t),\mathbf{u}(t)) \tag{4.41}$$

である. ここで, $\bar{\boldsymbol{\phi}}(\mathbf{s}(t+1),\mathbf{u}(t+1)) = [0,\ldots,0,\phi_{n+1}(\mathbf{s}(t+1)),\ldots,\phi_m(\mathbf{s}(t+1))]'$ となるベクトルである. よって, 二乗誤差 (4.32) を最小にするパラメータ $\mathbf{w}$ は

$$\mathbf{w} = \mathbf{A}^{-1}\mathbf{B} \tag{4.42}$$

$$\mathbf{A} = \alpha(T)\sum_{t=0}^{T}\left(\prod_{\tau=t+1}^{T}\beta(\tau)\right)\boldsymbol{\varphi}(t)\boldsymbol{\varphi}'(t) \tag{4.43}$$

$$\mathbf{B} = \alpha(T)\sum_{t=0}^{T}\left(\prod_{\tau=t+1}^{T}\beta(\tau)\right)\boldsymbol{\varphi}(t)r(\mathbf{s}(t),\mathbf{u}(t)) \tag{4.44}$$

と計算できる. 正規化項 $\alpha(T)$ は反復計算 $\alpha(t) = \alpha(t-1)/\left(\beta(t) + \alpha(t-1)\right)$ によって求めることができるため, 重み付き平均 $\langle\!\langle f(\cdot)\rangle\!\rangle(T) \equiv \alpha(T)\sum_{t=1}^{T}\left(\prod_{\tau=t+1}^{T}\beta(\tau)\right)f(\cdot)$, も

$$\langle\!\langle f(\cdot)\rangle\!\rangle(t) = (1-\alpha(t))\langle\!\langle f(\cdot)\rangle\!\rangle(t-1) + \alpha(t)f(\cdot) \tag{4.45}$$

のように反復計算によって求められる.

4.4.3 エピソード学習

学習のはじめに, actor のパラメータは ランダムに初期化され, actor のパラメータが最後に更新された時刻を保存する変数 $t_{\text{last-update}}$ を 0 に設定する. そして, 後述する学習エピソードが繰り返される.

各々の学習エピソードのはじめに物理システムと基本 CPG が初期状態に初期化され, t_0 を 0 に設定する. 時刻 t において actor は CPG 結合システムの状態 $\mathbf{s}(t)$ を観測し, 方策 $\pi_{\boldsymbol{\theta}}$ に従い間接制御信号 $\mathbf{u}(t)$ を出力する. その後, CPG 結合システムはシステムのダイナミクス (式 (2.1)〜(3.6)) に従い状態を $\mathbf{s}(t)$ から $\mathbf{s}(t+1)$ へと変化させる. 同時に critic は直接報酬 $r(\mathbf{s}(t), \mathbf{u}(t))$ を観測し, 重み付き十分統計量

$$\begin{aligned}\mathbf{A} &= \left\langle\!\left\langle \boldsymbol{\varphi}(t)\boldsymbol{\varphi}'(t)\right\rangle\!\right\rangle \\ \mathbf{B} &= \langle\!\langle \boldsymbol{\varphi}(t)r(\mathbf{s}(t), \mathbf{u}(t))\rangle\!\rangle\end{aligned} \tag{4.46}$$

を更新する. 1 エピソード内では, 以上の過程が時刻 $t - t_0 = t_{max}$ となるか, 物理システムが終了状態になるまで繰り返される. この間 actor のパラメータは固定される.

1 エピソードが終了し, $t - t_{\text{last-update}}$ が actor と critic のパラメータが固定される間隔 N よりも大きい場合, critic のパラメータ $\mathbf{w}$ は

$$\mathbf{w} = \mathbf{A}^{-1}\mathbf{B}, \tag{4.47}$$

のように再推定され, actor のパラメータが

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \eta_a \mathbf{w} \tag{4.48}$$

のように更新される. ここで, η_a は actor の学習係数である. これらのパラメータの更新が行われた場合, $t_{\text{last-update}}$ が t に設定される. 一方, $t - t_{\text{last-update}} \leq N$ の場合にはパラメータの更新は行われない.

強化学習は以上の過程を繰り返して行われる．この方法では, actor と critic 両方のパラメータを各時刻においてオンライン的に学習することも可能である．しかし, actor は現在の critic に基づいて学習を行い, critic は現在の actor (方策) の下での価値関数を推定するというように actor と critic は相互に依存している．Critic のパラメータが収束するのは actor が固定でサンプルが無限にあるときであり, actor のパラメータが収束するのは critic のパラメータが式 (4.25) を最小にする場合である．学習過程を安定にするため, actor と critic のパラメータの更新は一定数 (N 個) のサンプルデータが得られるまで行われない．

4.5. 実験:自然方策勾配法

自然方策勾配法に基づく学習法を 4.3 節と同様に 2 足歩行ロボットシミュレータ [10] に適用した．シミュレーション実験の目的は, ロボットが安定した歩行運動を行うような CPG コントローラのパラメータを自律的に獲得することである．

実験条件

Actor の方策 $\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})$ を

$$\pi_\theta(\mathbf{u}|\mathbf{s}) = \mathcal{N}(\bar{\mathbf{u}}, \sigma) = (2\pi)^{-1/2}\sigma^{-1}\exp\left\{-\frac{(\mathbf{u}-\bar{\mathbf{u}})'(\mathbf{u}-\bar{\mathbf{u}})}{2\sigma^2}\right\} \tag{4.49}$$

とした．ただし, $\bar{\mathbf{u}}$ は式 (4.18) で与えられるものとする．また, σ は方策のランダムさを決める分散パラメータで, 定数とした．

報酬関数は 4.3 節の報酬から変更を加え,

$$\tilde{r}(\mathbf{x}) = k_h r_h(\mathbf{x}) + k_s r_s(\mathbf{x}) + k_{h2} r_{h2}(\mathbf{x}) + k_{\dot{h}} r_{\dot{h}}(\mathbf{x}) \tag{4.50}$$

$$r_h(\mathbf{x}) = h_1 - \min(h_4, h_5) - H$$

$$r_s(\mathbf{x}) = \begin{cases} x_7 & \text{if } |x_7| < 1 \\ x_7/|x_7| & \text{otherwise} \end{cases}$$

$$r_{h2}(\mathbf{x}) = \begin{cases} -r_h^2 & \text{if } r_h < 0 \\ 0 & \text{otherwise} \end{cases}$$

$$r_{\dot{h}}(\mathbf{x}) = \begin{cases} \exp(x_8 - \dot{H}) - 1 & \text{if } x_8 < \dot{H} \\ 0 & \text{otherwise} \end{cases},$$

とした. ここで, $h_i (i = 1, 4, 5)$ link-i の高さを示しており, x_7 と x_8 はそれぞれ link-1 の水平と垂直方向に対する速度を示している. $r_h(\mathbf{x})$ と $r_s(\mathbf{x})$ は 4.3 節のものと同様に転倒しないこと, また前に進むことに対する報酬で, r_{h2} と $r_{\dot{h}}$ は転倒した場合にさらに大きな罰を加えるための報酬である. これらの罰は安定な歩行を行っている場合にはほとんど加えられることはない. k_h, k_s, k_{h2} と $k_{\dot{h}}$ はそれぞれ r_h, r_s, r_{h2} と $r_{\dot{h}}$ に対する重みで, H と $\dot{H}$ は腰の高さと link-1 の垂直方向の速度の閾値である. また, これらのパラメータは $k_h = 1.0$, $k_s = 0.1$, $k_{h2} = 1.0$, $k_{\dot{h}} = 1.0$, $H = 0.9$ と $\dot{H} = -0.8$ のように設定した.

4.3 節と同様に各エピソードのロボットの初期状態は脚をわずかに開いて静止した状態, ニューロンの初期状態は全てのニューロンの内部状態を 0 とした. また, 初期状態の脚の角度はエピソード毎に一定の角度内でランダムに異なる角度を選んだ. 強化学習はこのような学習手続きを繰り返すことによって行われた.

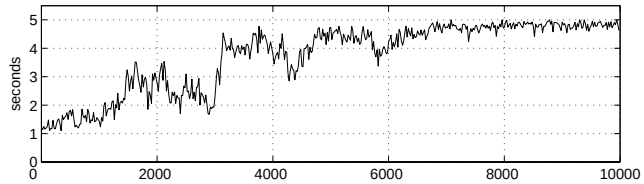
critic の 1〜8 番までの基底関数は方策パラメータ $\boldsymbol{\theta}$ から自動的に決まる $\psi_i = \phi_{\theta_i} = \nabla \ln \pi_\theta(\mathbf{u}|\mathbf{s})$ とし, その他に定数 : $\phi_9 \equiv 1$, link-1 の高さ: $\phi_{10}(t) \equiv x_2(t) - 0.9$, link-1 の水平方向の速度: $\phi_{11}(t) \equiv x_7(t)$, と link-1 の垂直方向の速度: $\phi_{12} \equiv x_8(t)$ の基底関数を加え, 12 個の基底関数を用いた.

4.5.1 歩行運動の獲得

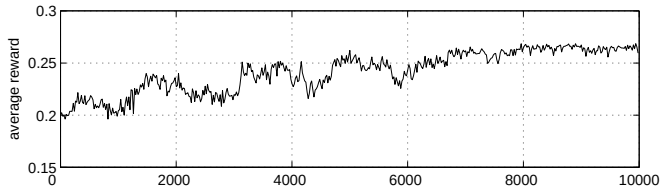
本実験では, 提案手法により 2 足歩行運動を可能にする CPG コントローラが学習によって得られるかを確かめる. Actor パラメータ $\boldsymbol{\theta}$ の初期値としてベクトルの各成分を

小さな乱数で与えた. このパラメータでは, 図 4.6(a) に示すように 2 足歩行ロボットはすぐに転倒する. また, 式 (4.19) の重み k_h と k_s をそれぞれ 1.0 と 0.1 とし, しきい値 H を 0.9 とした. ロボットが転倒した場合に大きな罰を与えるために, 式 (4.19) に r_h が 0 以下となった場合に $-r_h^2$ という負の報酬を追加した.

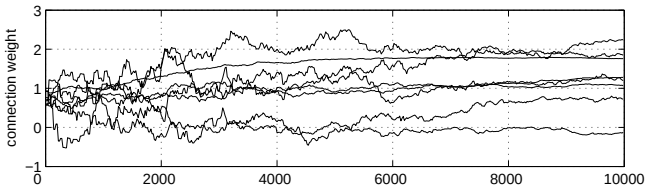
1 エピソードは最大 5 秒間とし, 5 秒以内にロボットが転倒した場合はその時点でエピソードを終了させる. また, actor と critic のパラメータの学習間隔 N は 1000 とした.



(a) 転倒するまでの時間



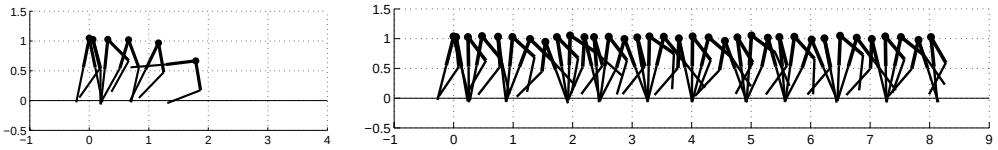
(b) 平均報酬



(c) Actor パラメータの変化

図 4.5 学習曲線

図 4.5 に学習曲線を示す. 軸は学習エピソード数を示し, 縦軸に学習エピソード内で転倒するまでの時間と 1 エピソード内で得られた報酬の平均をそれぞれ図 4.5(a) と図 4.5(b) に示す. また, これらの図は 20 エピソード分の移動平均である. また, 図 4.5(c) に方策パラメータ $\boldsymbol{\theta}$ の変化を示す. 約 7000 学習エピソード後に, ロボットは 5 秒間の間に転倒することが無くなり, 得られる報酬が大きくなっていることが分かる. 図 4.6 に 10000 学習エピソード後のロボットの歩容を示す. また, この時の方策パラメータは $\boldsymbol{\theta}_{npg1} = \{1.27, -0.12, 2.26, 1.07, 0.73, 1.17, 1.86, 1.76\}$ となった.



(a) 学習前の歩容　　(b) 10000 学習エピソード後の歩容

図 4.6 歩容

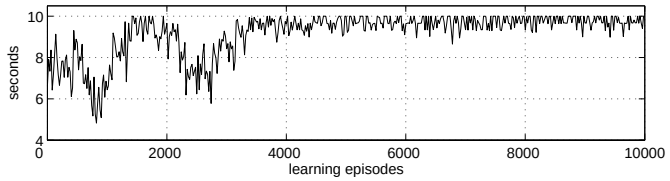
4.5.2 不整地上での学習

次に, 提案手法により不整地においても安定した歩行を生成できる CPG コントローラが獲得できるかを調べた. actor のパラメータ $\boldsymbol{\theta}$ の初期値を文献 [10] で用いられた値 $\boldsymbol{\theta}_{HT} = \{1.50, 1.00, 1.50, 1.50, 3.00, 1.50, 3.00, 1.50\}$ とし, 学習を行った. 図 4.8(a) に学習前のパラメータ $\boldsymbol{\theta}_{HT}$ を用いたときの不整地上での歩容を示す. 各学習エピソードの初めに, 地面を平均 1 m の区間に区切り, 区間毎にランダムに勾配 (正接が -0.05 〜 0.05 の範囲) を設定した. ただし, ロボットが歩き始める位置付近は平坦とした. 1 エピソードは最大 10 秒間とし, 10 秒以内にロボットが転倒した場合はその時点でエピソードを終了させる. また, actor と critic のパラメータの更新間隔 N は 5000 とした.

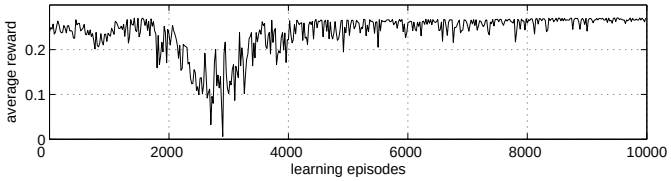
図 4.7 に学習曲線を示す. 軸は学習エピソード数を示し, 縦軸に学習エピソード内で転倒するまでの時間と 1 エピソード内で得られた報酬の平均をそれぞれ図 4.7(a) と図 4.7(b) に示す. また, これらの図は 20 エピソード分の移動平均である. 約 4000 学習エピソード後に不整地においてもロボットがほとんど転倒しないような良い制御則を獲得した. 図 4.8(b) に 10 000 学習エピソード後のロボットの歩容を示す. また, 10000 学習エピソード後の感覚フィードバック結合の重みは $\boldsymbol{\theta}_{npg2} = \{1.75, 0.12, 3.55, 1.74, 3.15, 1.40, 2.44, 1.47\}$ であった.

4.6. 学習によって得られたパラメータの性能評価

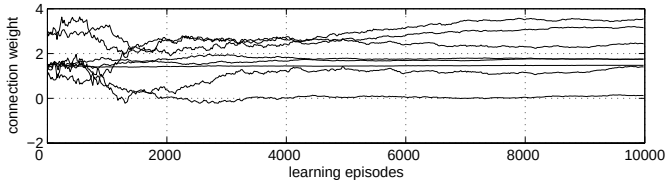
学習で得られた方策パラメータの性能を評価するため, 平坦地, 下り (急・緩) 勾配, 上り (急・緩) 勾配と不整地の複数の地面の条件の下で制御タスクを行った. 各制御タスクのロボットの初期状態は学習で用いた静止姿勢とし, 20 秒間の制御を行った. 文献 [10] で用いられた方策パラメータ $\boldsymbol{\theta}_{HT}$, 方策勾配法によって獲得した方策パラメータ $\boldsymbol{\theta}_{pg1}$ と $\boldsymbol{\theta}_{pg2}$, また自然方策勾配法によって獲得した方策パラメータ $\boldsymbol{\theta}_{npg1}$ と $\boldsymbol{\theta}_{npg2}$, さらに 3 章で得られた結合重みパラメータ $\mathbf{a}_{RL}$ のそれぞれに対し, 制御タスクを 50 回繰り返した. 表



(a) 転倒するまでの時間



(b) 平均報酬

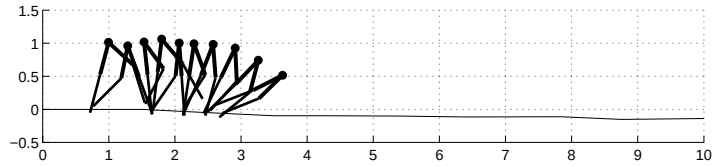


(c) Actor パラメータの変化

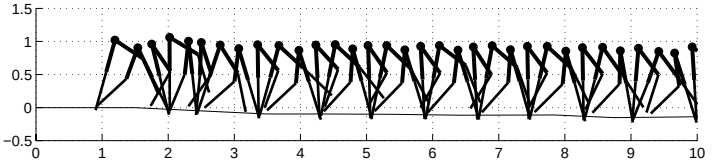
図 4.7 学習曲線

4.1 に制御タスク中の 20 秒間にロボットが転倒しなかった回数を示す. 図 4.9 〜 4.12 に方策パラメータ $\boldsymbol{\theta}_{pg1}$, $\boldsymbol{\theta}_{pg2}$, $\boldsymbol{\theta}_{npg1}$ と $\boldsymbol{\theta}_{npg2}$ を用いたときの下り (急・緩) 勾配, 上り (急・緩) 勾配と不整地での歩容を示す. 上り (急), 上り (緩), 下り (緩) と下り (急) の勾配はそれぞれ正接が 0.1, 0.05, -0.05 と 0.1 となる勾配で, 不整地のモデルは 4.5.2 節と同様に生成した. ただし, 設定される勾配の範囲は正接が -0.1 から 0.1 の範囲とした.

結果より, 学習によって得られた CPG コントローラにより地面の条件が異なる場合にも安定した歩行を生成できることが分かる. 転倒回数で比較すると 3 章で得られた CPG コントローラ $\mathbf{a}_{RL}$ と自然方策勾配法による $\boldsymbol{\theta}_{npg2}$ の成績が良い. また, $\boldsymbol{\theta}_{HT}$ を平坦な地面上で調整した $\boldsymbol{\theta}_{pg2}$ は $\boldsymbol{\theta}_{HT}$ よりも全ての項目で上回っており, 学習による性能の向上が確認できる. さらに, 不整地上で学習を行った $\boldsymbol{\theta}_{npg2}$ が安定した 2 足歩行を生成しており, 提案手法を用いた学習によって環境に適応できることが分かる. 平坦な地面で学習を行った場合にも, 文献 [10] で用いられた方策パラメータ $\boldsymbol{\theta}_{HT}$ よりも転倒回数が減少している理由としては, 最適方策を探索するために学習時に加えられるノイズが外乱のように働くため, 外乱が働いた場合にも転倒しないような方策を学習していることが考えられる.



(a) 学習前



(b) 学習後

図 4.8 不整地での歩容

表 4.1 学習によって得られた方策パラメータの性能比較

	平坦	下り (急)	下り (緩)	上り (緩)	上り (急)	不整地
$\boldsymbol{\theta}_{HT}$	46	22	44	10	0	2
$\boldsymbol{\theta}_{pg1}$	45	26	31	25	21	9
$\boldsymbol{\theta}_{pg2}$	49	48	50	40	1	4
$\boldsymbol{\theta}_{npg1}$	50	49	47	36	18	27
$\boldsymbol{\theta}_{npg2}$	46	49	50	50	29	35
$\mathbf{a}_{RL}$	49	50	50	33	22	45

4.7. 考察

3 章で用いた価値関数に基づいた強化学習法で critic が Q 関数の近似を行い, actor が critic の出力に基づいて学習を行った. この手法では, CPG 結合システムが高次元の状態, 行動空間を持つために critic の近似する Q 関数は高次元の関数であった. 次元の増加と共に関数近似問題は困難になるにも関わらず, actor が critic の出力に基づいて学習を行うため, 強化学習タスクを成功させるためには, critic の行う近似が正確である必要がある.

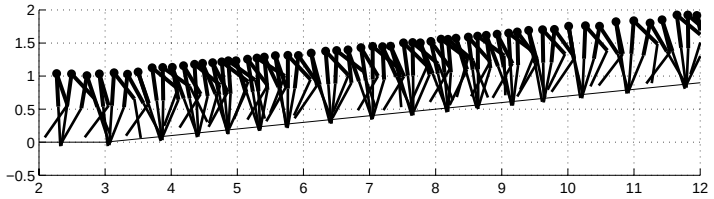
本章では, 確率的方策勾配法に基づいた学習手法を提案した. Critic は真の行動価値関数では無く低次元空間に射影された行動価値関数の学習を行い, actor は critic に基づいて学習を行う.

初めに, Konda らの actor-critic モデルに基づいた学習法を提案した. 提案手法を 2 足

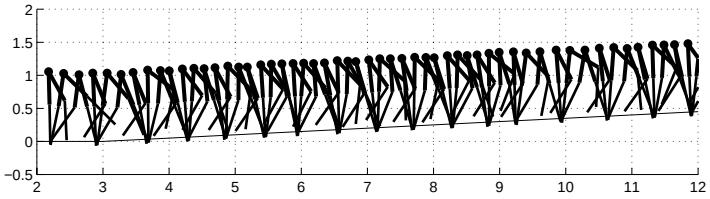
歩行運動の自律的な獲得課題に適用し, 提案手法により初期の方策パラメータ付近の局所最適なパラメータを学習できることを示した. しかしながら, 学習が遅いという問題があった.

次に, 学習を高速にするため自然方策勾配法に基づいた学習手法を提案した. 自然方策勾配法を用いることにより, 通常の方策勾配法に比べて少ない学習エピソード数で安定した歩行を実現する CPG コントローラを獲得することができた. さらに, 不整地上で学習を行うことにより, 文献 [10] で用いられたハンドチューニングによるパラメータでは安定した歩行が不可能な不整地上でも, 安定した歩行が可能な CPG コントローラを獲得することができた.

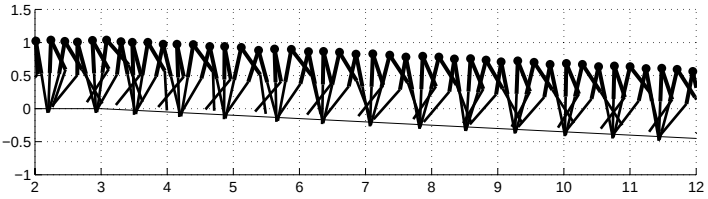
また, 本章で提案した手法によって得られた CPG コントローラと 3 章で得られたもの $\mathbf{a}_{RL}$, さらに文献 [10] で用いられたもの $\boldsymbol{\theta}_{HT}$ の性能を比較した. $\mathbf{a}_{RL}$ と同程度の性能の CPG コントローラが得られることを示した. これらのコントローラはハンドチューニングの $\boldsymbol{\theta}_{HT}$ に比べると良い成績であった. しかしながら, 方策勾配法に基づく手法を用いた場合の学習過程は安定したものとなっており, 特に次元の高い状態行動空間を持つ 2 足歩行運動の獲得課題などに対しては, 価値関数に基づく手法に比べて方策勾配法が有効であると考えられる.



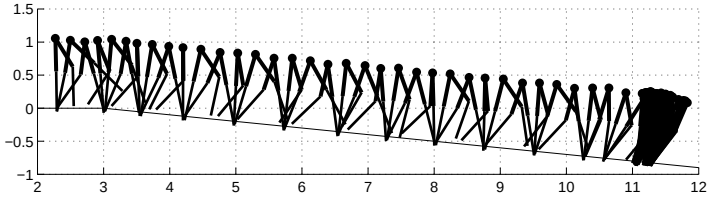
(a) 上り (急)



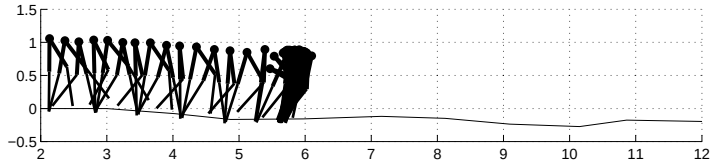
(b) 上り (緩)



(c) 下り (緩)

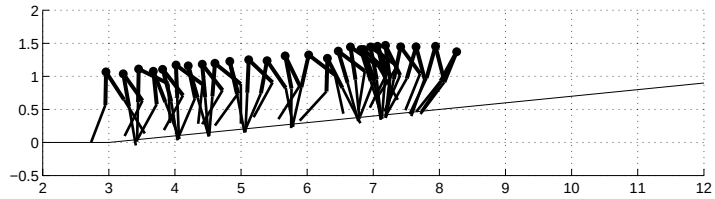


(d) 下り (急)

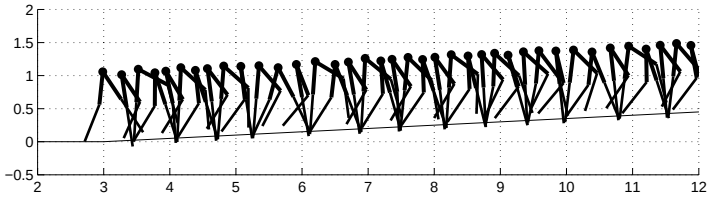


(e) 不整地

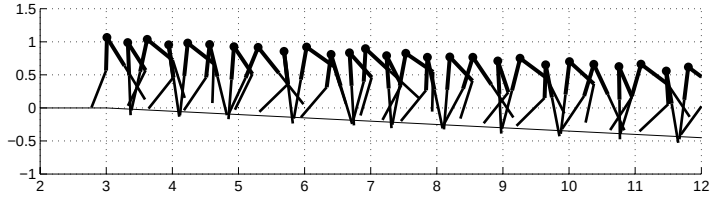
図 4.9 $\boldsymbol{\theta}_{pg1}$ を用いた場合の歩容



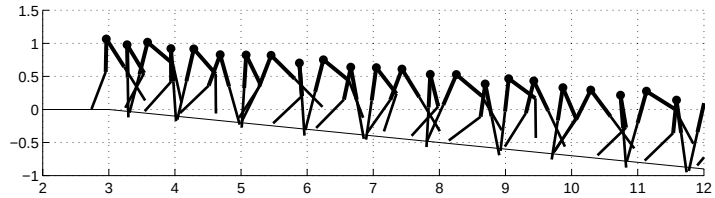
(a) 上り (急)



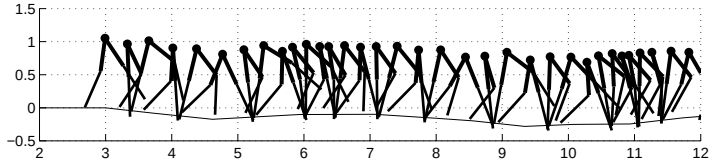
(b) 上り (緩)



(c) 下り (緩)

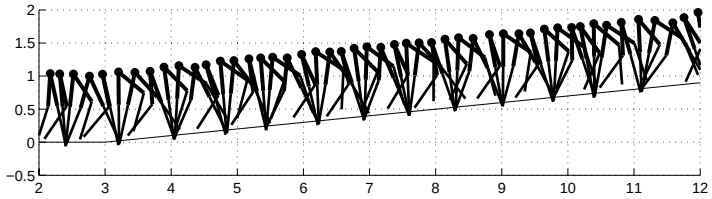


(d) 下り (急)

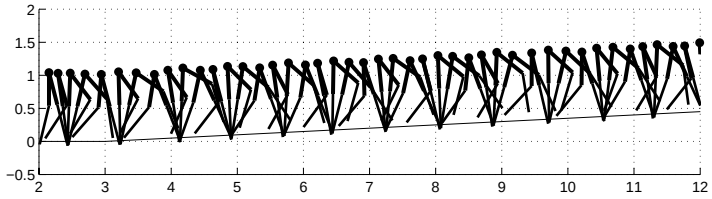


(e) 不整地

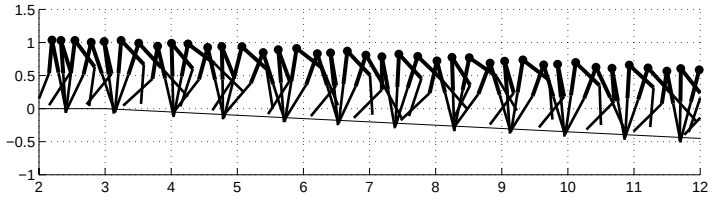
図 4.10 $\boldsymbol{\theta}_{pg2}$ を用いた場合の歩容



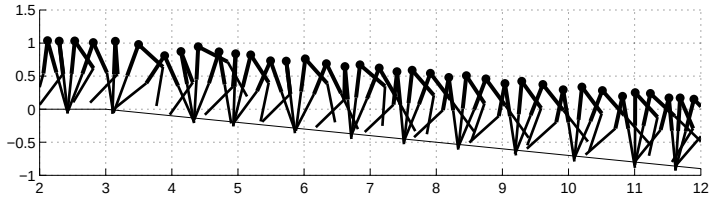
(a) 上り (急)



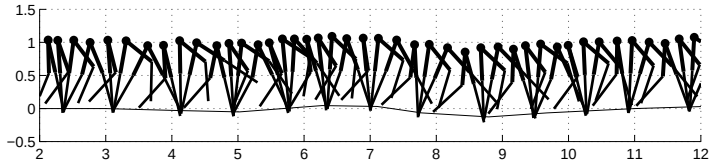
(b) 上り (緩)



(c) 下り (緩)

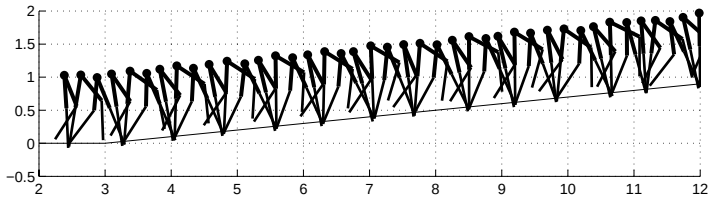


(d) 下り (急)

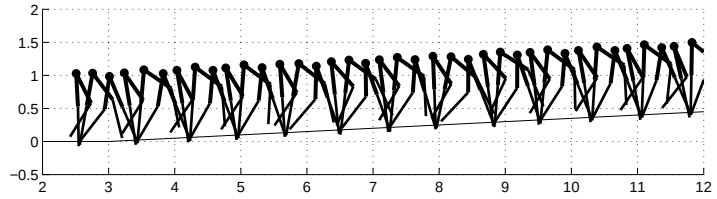


(e) 不整地

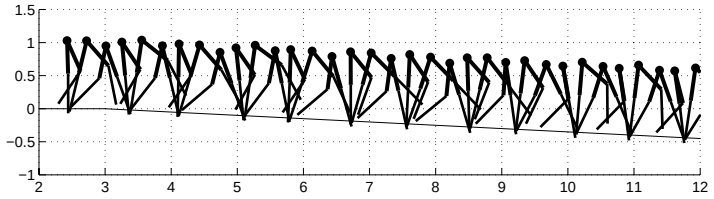
図 4.11 $\boldsymbol{\theta}_{npg1}$ を用いた場合の歩容



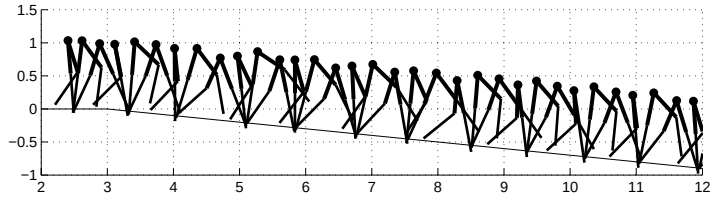
(a) 上り (急)



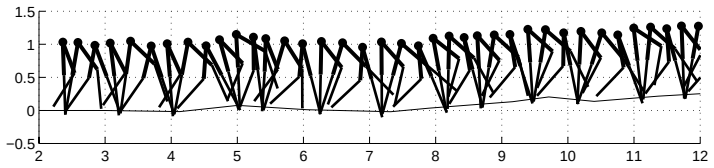
(b) 上り (緩)



(c) 下り (緩)



(d) 下り (急)



(e) 不整地

図 4.12 $\boldsymbol{\theta}_{npg2}$ を用いた場合の歩容

第5章　まとめ

5.1. 議論

CPG コントローラを用いて物理システムの制御を行う場合, 出力信号が周期的であるという CPG コントローラの持つ特性により, 物理システムに対する制御トルクのパターンが限られたものとなり, 取り得る状態空間が経路状の小さな空間に制限されることが期待される. 3 章で用いた NGnet などの局所モデルを用いて価値関数の近似を行う場合, 縮退した空間内において価値関数の学習を行えば良いと考えられるが, 価値関数に基づく学習法では学習過程が不安定であった. これは状態行動空間自体は小さな空間に縮退しているが, 価値関数の入力次元は元の状態行動空間の持つ次元と同じであり, 特に CPG-actor-critic モデルでは CPG ニューロンの状態も制御対象の状態として扱うため, 本来の制御対象である物理システムの状態空間よりも次元の高い空間を扱う必要があることが理由として考えられる. この問題に対する改善方法として, 観測する CPG ニューロンの数を制限することが有効であると考えられる. 観測しない状態変数が存在すると部分観測問題の難しさを生じさせるが, CPG は周期的なパターンを生成するものであり, CPG の持つ状態空間が全ニューロンの内部状態で表されるベクトル空間内の小さな空間に縮退していると考えられるからである.

本論文では, 価値関数に基づく学習法と方策勾配法に基づく学習法を提案したが, これらの手法では目的関数が異なっている. 価値関数に基づく学習法は各状態と行動の組に対して累積報酬を予測し, 各状態においてより良い行動を出力するように学習を行うのに対し, 方策勾配法に基づく学習法では, 定常分布の下での平均報酬または累積報酬の期待値を大きくするように方策パラメータの学習を行う. すなわち, 価値関数法では各時刻に訪れた状態において局所的に評価の高い行動を学習するのに対して, 方策勾配法では各状態に訪れる頻度により重みづけされた大域的に評価の高い方策を学習するという違いがある. しかしながら, 本論文で扱った CPG コントローラの学習においては, 方策自体が大域的であり, 価値関数法においても方策パラメータの更新回数は各状態を訪れる頻度に

比例するため, 結果的に各状態において最適方策を探索するのではなく, 大域的に評価の高い方策を探索していると考えられる.

また, 方策勾配法では定常分布の下で得られる報酬の最大化として定式化されるため, エピソード学習を行う場合には各学習エピソードにおける初期状態の設定に関する問題がある. すなわち, 4 章の実験では, 各エピソードにおける CPG 結合システムの初期状態として, 2 足歩行ロボットは静止姿勢, CPG ニューロンの状態は 0 としたが, このような初期状態を設定し, ロボットの転倒, または一定時間の結果によってエピソードを終了する場合, 一般に経験分布が方策 $\pi_{\boldsymbol{\theta}}$ の下での定常分布 $D_{\boldsymbol{\theta}}$ とは異なる分布となる. この経験分布と定常分布の違いにより方策勾配の推定値にバイアスが加わり, 学習に悪影響を与える要因となる. また, 価値関数法においても, 歩き始めと歩行中の最適方策が異なる場合には 1 エピソードの最大時間などの終了条件によって目的関数が変わるため, 学習エピソードの初期状態の設定や終了条件についても研究を行う必要がある.

5.2. まとめ

本論文では, 生物の制御機構を模倣した CPG コントローラを用いたリズム運動制御を自律的に獲得するための新しい強化学習法である CPG-actor-critic モデルを提案し, 感覚フィードバック結合の重み W^{act} と CPG ニューロン間の相互結合重み W^{feed} に対する学習を行う手法を導出した. CPG コントローラのパラメータにはそれらの他に CPG ニューロンのバイアス入力 B_i と時定数 c_i, 物理システムに対する制御トルクの重み T_i がある. Actor がこれらのパラメータを含むように拡張することで, W^{act} や W^{feed} と同様に学習が可能である. しかし, CPG コントローラを用いて制御を行う場合, 物理システムの運動が CPG 固有のリズムに引き込まれるという特性を利用しているため, 神経振動子内のパラメータ全てを可変にすることは望ましくない. また, 結合重み W^{act} や W^{feed} の学習を行うとニューロンの出力が変化するため, 重み $\mathbf{T}$ を固定しても制御トルク $\boldsymbol{\tau}$ は変化する. よって, これらのパラメータに対する学習を行う必要性は少ない.

3 章では, 価値関数に基づく CPG-actor-critic モデルの学習法を導出し, 提案手法を 2 足歩行ロボットシミュレータに対する CPG コントローラの獲得課題に適用した. シミュレーションの結果, 安定な 2 足歩行を実現する CPG コントローラが獲得できることを示した. また, 学習によって得られた CPG コントローラを用いた場合, 登り勾配, 下り勾配やデコボコ道においても安定した歩行が可能であった. しかしながら, 学習の過程が不安

定であった．学習過程の不安定さは価値関数の学習が困難であることが理由として考えられる．

4 章では, 3 章で示した価値関数に基づく手法の問題点である学習過程の不安定さを改善するため, 確率的方策勾配法に基づく CPG-actor-critic モデルの学習法を導出した．提案手法は, Konda らに提案された actor-critic 法に基づいており, 真の評価関数ではなく, 学習の容易な低次元空間へ写像した評価関数を用いて学習できる [35]．また, 3 章と同様に 2 足歩行ロボットシミュレータに対する CPG コントローラの獲得課題に適用し, 学習初期のパラメータ付近の局所最適なパラメータに収束することが示唆された．大域的に最適なパラメータを探索する手法が必要であると考えられる．また, 学習が遅く, 学習速度について改善する必要があった．そこで, 自然方策勾配法に基づいた学習手法を提案し, 少ない学習エピソード数で安定した歩行を実現する CPG コントローラが獲得できることを示した．また, 不整地上で学習を行うことにより, 提案手法を用いた学習によって新しい環境に適応できることを示した．

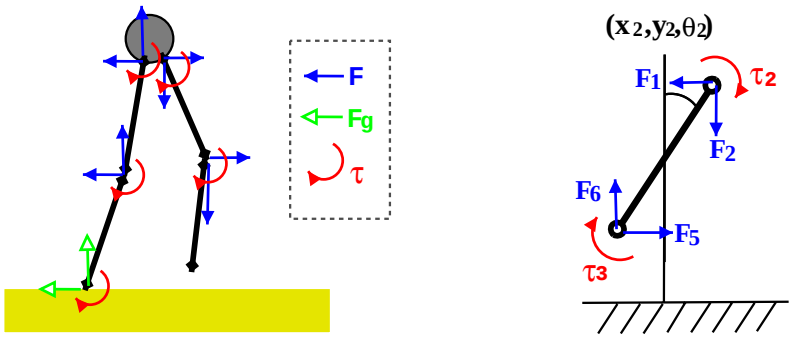
本論文で行った実験では CPG ニューロン間の結合重みの学習は行わなず, 感覚フィードバック結合の重みだけを学習した．提案手法を用いて原理的には神経振動子間の結合パラメータの学習も可能である．しかし, 今回の 2 足歩行の制御タスクでは結合重みが多数あり, 学習を行うパラメータ数が増加すると学習は難しくなる．また, CPG を用いた制御法は CPG が生得的に発生する信号パターンを利用した制御方法であり, 自由に変化させることは長所を失わせることになる．そこで本論文では調整できるパラメータに制限を加えた比較的容易な実験により, 提案手法を用いた CPG コントローラの学習が可能であることを示した．しかし, CPG 内の結合重みを変化させることにより, より良い制御が獲得できると考えられるため, CPG-actor-critic モデルを用いて CPG 内の結合重みの調整を行うことは今後の課題である．

付録

A. 2 足歩行ロボットシミュレータ

本節では, Taga ら [10] で用いられた 2 足歩行ロボットシミュレータの実装について説明する.

A.1 2 足歩行ロボットのダイナミクス計算



(a) ロボットに加わる力　　(b) link-2 に加わる力

図 5.1 剛体リンクに加わる力

ロボットは腰から上の体を代表した質点, 左右の股, 左右の脛と左右の足で構成される. 足は非常に小さいものとし, ダイナミクスの計算では無視するが, かかとには接地時に限りトルクを加えることができる.

図 5.1(a) に示すように, 2 足歩行ロボットには 3 種類の力が加わる. $\mathbf{F}$ は各リンクが結合しているために生じる運動学的な力である. 結合部毎に 2 方向 (鉛直, 水平方向) の力が加わるため, 全体で 8 次元ベクトルの力である. また, $\mathbf{F_g}$ は地面から加えられる力で, 足が接地している場合に鉛直, 水平方向の力が加わる. そして, $\boldsymbol{\tau}$ は制御トルクで各関節に加えられる. 足が接地している場合にはかかとにトルクが加えられる.

link-2 の状態は水平と垂直位置, 鉛直軸からの回転角の 3 次元ベクトル (x_2, y_2, θ_2) で

表され, 加わる力は図 5.1(b) に示すようになる. 他のリンクに対しても同様な手続きを行うことで全リンクの運動方程式を求めると

$$
\begin{aligned}
M\ddot{x}_1 &= F_1 + F_3 \\
M\ddot{y}_1 &= F_2 + F_4 - Mg \\
m_1\ddot{x}_2 &= -F_1 + F_5 \\
m_1\ddot{y}_2 &= -F_2 + F_6 - m_1 g \\
i_1\ddot{\theta}_2 &= -F_1 l_1 \cos\theta_2 + F_2 l_1 \sin\theta_2 - F_5 l_1 \cos\theta_2 + F_6 l_1 \sin\theta_2 \\
&\quad - b_1|\theta_2|\dot{\theta}_2 - \{b_2 + b_k f(\theta_2 - \theta_4)\}(\dot{\theta}_2 - \dot{\theta}_4) + \tau_1 + \tau_3 \\
m_1\ddot{x}_3 &= -F_3 + F_7 \\
m_1\ddot{y}_3 &= -F_4 + F_8 - m_1 g \\
i_1\ddot{\theta}_3 &= -F_3 l_1 \cos\theta_3 + F_4 l_1 \sin\theta_3 - F_7 l_1 \cos\theta_3 + F_8 l_1 \sin\theta_3 \\
&\quad - b_1|\theta_3|\dot{\theta}_3 - \{b_2 + b_k f(\theta_3 - \theta_5)\}(\dot{\theta}_3 - \dot{\theta}_5) + \tau_2 + \tau_4 \\
m_2\ddot{x}_4 &= -F_5 + Fg_1 \\
m_2\ddot{y}_4 &= -F_6 + Fg_2 - m_2 g \\
i_2\ddot{\theta}_4 &= -F_5 l_2 \cos\theta_4 + F_6 l_2 \sin\theta_4 - Fg_1 l_2 \cos\theta_4 + Fg_2 l_2 \sin\theta_4 \\
&\quad - \{b_2 + b_k f(\theta_2 - \theta_4)\}(\dot{\theta}_4 - \dot{\theta}_2) + k_k h(\theta_2 - \theta_4) - \tau_3 - \tau_5 \\
m_2\ddot{x}_5 &= -F_7 + Fg_3 \\
m_2\ddot{y}_5 &= -F_8 + Fg_4 - m_2 g \\
i_2\ddot{\theta}_5 &= -F_7 l_2 \cos\theta_5 + F_8 l_2 \sin\theta_5 - Fg_3 l_2 \cos\theta_5 + Fg_4 l_2 \sin\theta_5 \\
&\quad - \{b_2 + b_k f(\theta_3 - \theta_5)\}(\dot{\theta}_5 - \dot{\theta}_3) + k_k h(\theta_3 - \theta_5) - \tau_4 - \tau_6
\end{aligned}
\tag{5.1}
$$

となる. ただし, $(\dot{})$ と $(\ddot{})$ はそれぞれ 1 階と 2 階の時間微分で, $f(x) = \max(0, x)$, $h(x) = 0$ if $x \leq 0$, 1 if $x > 0$ である. また, M は link-1 の重さであり, m_1, i_1, l_1 はそれぞれ link-2, 3 の重さ, モーメント, 長さ m_2, i_2, l_2 はそれぞれ link-4, 5 の重さ, モーメント, 長さを表す. また, 各リンクは密度が均一の棒状の剛体で, モーメントは $i = ml^2/12$ と計算される. ここで $\mathbf{x} = \{x_1, y_1, x_2, \ldots, y_5, \theta_5\}$ と書き, 式 (5.1) を $\mathbf{F}$ に対してまとめると

$$
\ddot{\mathbf{x}} = P(\mathbf{x})\mathbf{F} + \mathbf{Q}(\mathbf{x}, \dot{\mathbf{x}}, \boldsymbol{\tau}, \mathbf{F_g}) \tag{5.2}
$$

と書ける. ここで $\mathbf{P}(\mathbf{x})$ は 14×8 の行列で, $\mathbf{Q}$ は 14×1 のベクトルである.

また, 各リンクが結合している条件から,

$$
\begin{aligned}
&x_1 = x_2 + l_1 \sin\theta_2 \\
&y_1 = y_2 + l_1 \cos\theta_2 \\
&x_2 - l_1 \sin\theta_2 = x_4 + l_2 \sin\theta_4 \\
&y_2 - l_1 \cos\theta_2 = y_4 + l_2 \cos\theta_4 \\
&x_1 = x_3 + l_1 \sin\theta_3 \\
&y_1 = y_3 + l_1 \cos\theta_3 \\
&x_3 - l_1 \sin\theta_3 = x_5 + l_2 \sin\theta_5 \\
&y_3 - l_1 \cos\theta_3 = y_5 + l_2 \cos\theta_5
\end{aligned}
\tag{5.3}
$$

を満たす必要がある. 式 (5.3) を 2 階微分して $\mathbf{x}$ を用いてベクトル表記すると

$$
C(\mathbf{x})\ddot{\mathbf{x}} = \mathbf{D}(\mathbf{x}, \dot{\mathbf{x}}) \tag{5.4}
$$

となる. ここで, $C(\mathbf{x})$ は 8×14 行列で, $\mathbf{D}(\mathbf{x}, \dot{\mathbf{x}})$ は 8 次元ベクトルである.

式 (5.2) と式 (5.4) より $\mathbf{F}$ は

$$
\mathbf{F} = \left[C(\mathbf{x})P(\mathbf{x})\right]^{-1} \left[\mathbf{D}(\mathbf{x}, \dot{\mathbf{x}}) - C(\mathbf{x})\mathbf{Q}(\mathbf{x}, \dot{\mathbf{x}}, \mathrm{Tr}(\mathbf{y}), \mathbf{F_g}(\mathbf{x}, \dot{\mathbf{x}}))\right] \tag{5.5}
$$

と計算でき, $\mathbf{x}$ の 2 階微分は

$$
\begin{aligned}
\ddot{\mathbf{x}} = {} & P(\mathbf{x})\left[C(\mathbf{x})P(\mathbf{x})\right]^{-1} \left[\mathbf{D}(\mathbf{x}, \dot{\mathbf{x}}) - C(\mathbf{x})\mathbf{Q}(\mathbf{x}, \dot{\mathbf{x}}, \mathrm{Tr}(\mathbf{y}), \mathbf{F_g}(\mathbf{x}, \dot{\mathbf{x}}))\right] \\
& + \mathbf{Q}(\mathbf{x}, \dot{\mathbf{x}}, \mathrm{Tr}(\mathbf{y}), \mathbf{F_g}(\mathbf{x}, \dot{\mathbf{x}}))
\end{aligned}
\tag{5.6}
$$

と計算できる.

また, 地面から受ける力 $\mathbf{F_g}$ は

$$
Fg_1 = \begin{cases} -k_g(x_r - \hat{x}_{r0}) - b_g\dot{x}_r & y_r - y_g(x_r) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.7}
$$

$$
Fg_2 = \begin{cases} -k_g(y_r - \hat{y}_{r0}) - b_g f(\dot{y}_r) & y_r - y_g(x_r) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.8}
$$

$$Fg_3 = \begin{cases} -k_g(x_l - \hat{x}_{l0}) - b_g\dot{x}_l & y_l - y_g(x_l) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

$$Fg_4 = \begin{cases} -k_g(y_l - \hat{y}_{l0}) - b_g f(\dot{y}_l) & y_l - y_g(x_l) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

で与えられる. Fg_1 と Fg_2 は link-4 が地面から受ける力で, x_r, y_r, x_{r0}, y_{r0} は link-4 のかかと (下端) の水平・鉛直方向の位置, link-4 が着地した瞬間のかかとの水平・鉛直方向の位置である. Fg_3 と Fg_4 は同様に link-5 が地面から受ける力である.

また, link-1 の重さは 48.0 kg, link-2, 3 の重さと長さはそれぞれ 7.0kg と 0.5m で, link-4, 5 の重さと長さはそれぞれ 4.0kg と 0.6m である. ロボットの運動方程式 (5.1), $\mathbf{F_g}$ の計算 (5.7)〜(5.10) で用いた係数の値は, $k_g = 10000$, $b_g = 1000$, $k_k = 10000$, $b_k = 1000$, $b_1 = 10$, $b_2 = 10$ である.

以上のように各リンクのダイナミクスが計算されるが, リンクが結合している条件より, ロボットの状態変数は $\{x_1, y_1, \theta_2, \theta_3, \theta_4, \theta_5, \dot{x}_1, \dot{y}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5\}$ の 12 次元ベクトルとなる. 本文ではこの 12 次元ベクトルを 2 足歩行ロボットの状態変数 $\mathbf{x}$ として扱う.

A.2 神経振動子ネットワーク

Taga [10] らが用いた神経振動子ネットワークについて説明する.

主ニューロン:1 から 12 番のニューロン 主ニューロンのダイナミクスは

$$\begin{aligned} c_i\dot{\nu}_i &= -\nu_i \sum_{j=1}^{1} 2W_{ij}y_j - 2.5y_{i+12} + 5.5 + I_i^{ext} \\ y_i &= \max(0, \nu_i) \end{aligned} \tag{5.11}$$

である. 股関節を制御するニューロンの時定数は $c_{1\sim4} = 0.05$ で, 膝, かかとの制御を行うニューロンの時定数は $c_{5\sim12} = 0.60$ である. 神経振動子内の主ニューロン $(2i, 2i-1,\ i = 1, 2, \ldots, 6\ )$ 間の相互結合の重みは $W_{2i,2i-1} = W_{2i-1,21} = -2$, $i = 1, 2, \ldots, 6$ である. また, 左右の股関節に対する制御を行う神経振動子は出力が反位相となるように $W_{1,3} = W_{3,1} = W_{2,4} = W_{4,2} = -1$ とする. さらに, 股関節の制御を行うニューロンから膝, かかとの制御を行うニューロンに対する結合重みは $W_{6,1} = W_{10,1} = W_{6,2} = W_{10,2} = W_{8,3} = W_{12,3} = W_{8,4} = W_{12,4} = -1$ とした. I_i^{ext} として後述する感覚フィードバック信号が入力される.

副ニューロン:13 から 24 番のニューロン 副ニューロンのダイナミクスは

$$\begin{aligned} c_i\dot{\nu}_i &= -\nu_i + y_{i-12} \\ y_i &= \nu_i \end{aligned} \tag{5.12}$$

である. 股関節を制御するニューロンの時定数は $c_{13\sim16} = 0.025$ で, 膝, かかとの制御を行うニューロンの時定数は $c_{17\sim24} = 0.30$ である.

A.3 制御トルク

制御トルクは式 (5.13)

$$\begin{aligned} \tau_i &= -T_i^F y_{2i-1} + T_i^E y_{2i} \quad (i = 1, \ldots, 4) \\ \tau_i &= (-T_i^F y_{2i-1} + T_i^E y_{2i})\Xi_{i-1} \quad (i = 5, 6) \end{aligned} \tag{5.13}$$

で与えられる. 結合重みは $T_1^F = T_2^F = 15.0$, $T_1^E = T_2^F = 85.0$, $T_3^F = T_4^F = 15.0$, $T_3^E = T_4^F = 15.0$, $T_5^F = T_6^F = 100.0$, $T_5^E = T_6^F = 75.0$ である.

A.4 感覚フィードバック信号

感覚フィードバック信号は $\mathbf{X} = \{x_3,\, x_4,\, x_5\Xi_4,\, x_6\Xi_5,\, \Xi_4,\, \Xi_5,\, x_{11}\Xi_4,\, x_{12}\Xi_5\}$ である. ここで, $x_i, i = 3, \ldots, 6$ は link-$(i-1)$ の角度で, x_{11} と x_{12} はそれぞれ link-4 と link-5 の角速度である.

ニューロン i へ入力される外部入力 I_i^{ext} は感覚フィードバック信号の重み付き和で与えられる.

$$\begin{aligned} I_1^{ext} &= a_1X_1 - a_2X_2 + a_3X_3 + a_4X_6, \\ I_3^{ext} &= a_1X_2 - a_2X_1 + a_3X_4 + a_4X_5, \\ I_5^{ext} &= a_5X_4, \quad I_7^{ext} = a_5X_3, \\ I_9^{ext} &= -a_6X_3 - a_7X_4 - a_8X_7, \\ I_{11}^{ext} &= -a_6X_4 - a_7X_3 - a_8X_8, \\ I_{2i}^{ext} &= -I_{2i-1}^{ext} \quad \text{for } i = 1, \ldots, 6 \end{aligned} \tag{5.14}$$

文献 [10] では $\mathbf{a} = \{1.5,\, 1.0,\, 1.5,\, 1.5,\, 3.0,\, 1.5,\, 3.0,\, 1.5\}$ が用いられた.

B. NGnet の学習

NGnet に対する学習手法である オンライン EM アルゴリズム [31] について説明する. オンライン学習ではモデルパラメータ θ の推定値がデータが観測される度に変化する. t 番目のデータを観測した後のモデルパラメータ θ の推定値を $\theta(t)$ で表す. オンライン EM アルゴリズムは新しいデータが観測されたときに以下の E-step と M-step を行い, モデルパラメータ θ の推定値を更新するアルゴリズムである.

E-step モデルパラメータ $\theta(t-1)$ の下で t 番目に観測されたデータ $\{q(t), \mathbf{S}(t)\}$ に対してユニット m が選択される事後確率を計算する. ここで, $q(t)$ は NGnet に与えられる出力の教師信号である. 事後確率 $P_m(t)$ は

$$\begin{aligned} P_m(t) &\equiv P(m|\mathbf{S}(t), q(t), \theta(t-1)) \\ &= \frac{p(\mathbf{S}(t), q(t), m|\theta(t-1))}{\sum_{m'}^{M} p(\mathbf{S}(t), q(t), m'|\theta(t-1))} \end{aligned}$$

のように計算できる. $p(\mathbf{S}(t), q(t), m|\theta(t-1))$ の計算には式 (3.13) を用いる.

M-step 重み付き平均を用いて確率モデルのモデルパラメータは以下のように求めることができる.

$$\mu_m(t) = \frac{\langle\langle \mathbf{S} \rangle\rangle_m(t)}{\langle\langle 1 \rangle\rangle_m(t)} \tag{5.15}$$

$$\tilde{\Lambda}_m(t) = \frac{1}{1-\eta(t)} \left[\tilde{\Lambda}_m(t-1) - \frac{P_m(t)\tilde{\Lambda}_m(t-1)\tilde{\mathbf{S}}(t)\tilde{\mathbf{S}}'(t)\tilde{\Lambda}'_m(t-1)}{(\frac{1}{\eta(t)}-1) + P_m(t)\tilde{\mathbf{S}}'(t)\tilde{\Lambda}'_m(t-1)\tilde{\mathbf{S}}(t)} \right] \tag{5.16}$$

$$\tilde{\mathbf{K}}_m(t) = \tilde{\mathbf{K}}_m(t-1) + \eta(t) P_m(t) \left(q(t) - \tilde{\mathbf{K}}_m(t-1)\tilde{\mathbf{S}} \right) \tilde{\mathbf{S}}'\tilde{\Lambda}(t) \tag{5.17}$$

$$\sigma_m^2(t) = \frac{\langle\langle |q|^2 \rangle\rangle_m(T) - \mathrm{Tr}\left(\tilde{\mathbf{K}}_m \left\langle\left\langle \tilde{\mathbf{S}} q' \right\rangle\right\rangle_m(t) \right)}{D\langle\langle 1 \rangle\rangle_m(T)} \tag{5.18}$$

ここで, $\tilde{\mathbf{K}}_m$ は $(\mathbf{K}_m, b_m)$ で, $\tilde{\mathbf{S}}$ は $(\mathbf{S}', 1)'$ (プライム記号 $(')$ は転置) となる $N+1$ 次元のベクトルである. また, $\tilde{\Lambda}_m(t)$ は

$$\tilde{\Lambda}_m(t) \equiv \left[\left\langle\left\langle \tilde{\mathbf{S}}\tilde{\mathbf{S}}' \right\rangle\right\rangle_m(t) \right]^{-1}$$

となる共分散行列を計算するための補助変数である. $\Sigma_m^{-1}(t)$ は補助変数 $\tilde{\Lambda}_m(t)$ と以下の関係を満たす.

$$\tilde{\Lambda}_m(t)\langle\langle 1 \rangle\rangle_m(t) = \begin{pmatrix} \Sigma_m^{-1}(t) & \Sigma_m^{-1}(t)\mu_m(t) \\ \Sigma_m^{-1}(t)\mu_m(t) & 1 + \mu'_m(t)\Sigma_m^{-1}(t)\mu_m(t) \end{pmatrix} \tag{5.19}$$

また, 重み付き平均は

$$\langle\!\langle f(\mathbf{S},q)\rangle\!\rangle_m(T) = \eta(T)\sum_{t=1}^{T}\left(\prod_{s=t+1}^{T}\lambda(s)\right) f(\mathbf{S}(t),q(t))P_m(t) \quad (5.20)$$

のように計算する. ここで, $\lambda(t)\in[0,1]$ は古い推定値による効果を徐々に忘却するための忘却係数である. また, $\eta(T) = 1/\left(\sum_{t=1}^{T}\left(\prod_{s=t+1}^{T}\lambda(s)\right)\right)$ は正規化係数であり, $\eta(t) = (1+\lambda(t)/\eta(t-1))^{-1}$ と繰り返し計算で求めることができる. また, 式 (5.20) は以下のように逐次的な計算によって求めることができる.

$$\begin{aligned}\langle\!\langle f(\mathbf{S},q)\rangle\!\rangle_m(t) = {}& \langle\!\langle f(\mathbf{S},q)\rangle\!\rangle_m(t-1) \\ & + \eta(t)\left[f(\mathbf{S}(t),q(t))P_m(t) - \langle\!\langle f(\mathbf{S},q)\rangle\!\rangle_m(t-1)\right] \quad (5.21)\end{aligned}$$

ユニットの動的操作　局所モデルでは入力空間の次元の増加と共に, 関数近似に必要なユニット数が増加する. しかし, 現実のデータは入力空間の一部に局在することが多い. よって, 入力データの出現するところにユニットを配置すれば無駄がない. そこで, 以下のようなユニットの動的な操作を行う.

ユニットの生成: 新しいデータ $(\mathbf{S}(t),q(t))$ を観測した場合に, 確率 $\max_m p(\mathbf{S}(t),q(t)|m,\theta(t-1))$ は現在のモデルパラメータ $\theta(t-1)$ を用いた場合に, $(\mathbf{S}(t),q(t))$ が生成される確らしさをあらわす. そこで, この確率がある閾値よりも小さいとき, そのデータを説明可能なユニットを新たに生成する.

ユニットの削除: 混合比に対応する十分統計量 $\langle\!\langle 1\rangle\!\rangle_m(t)$ は t 番目のデータが観測されるまでに、ユニット m がどの程度用いられてきたかを示す。この十分統計量がある閾値よりも小さい場合には、このユニットがほとんど使われていないことを意味するので、そのユニットを削除する。

B.1 大域的ユニット

大域的ユニット m_g は, 入力データ $\mathbf{S}$ が大域ユニットから生成される確率 $p(\mathbf{S}|m_g,\theta)$ が定数となるユニットである. ここで, m_g は大域的ユニットのインデックスである. ある観測データ $\{\mathbf{S},q\}$ に対する大域ユニットの尤度 $p(\mathbf{S},q|m_g,\theta)$ も定数とする. また, 大域的ユニットの線形回帰行列は $\mathbf{K}_{m_g}\equiv 0$ とし, 出力は $q_{m_g} = b_{m_g}$ とした.

大域的ユニット m_g のモデルパラメータは

$$b_{m_g} = \langle\langle q \rangle\rangle_{m_g}(t)$$
$$\sigma^2_{m_g} = \left\langle\left\langle (q - b_{m_g}(t))^2 \right\rangle\right\rangle_{m_g} = \left\langle\left\langle q^2 \right\rangle\right\rangle_{m_g}(t) - b^2_{m_g}(t)$$

で求められる. 教師データ $\{\mathbf{S}(t), q(t)\}$ が与えられたとき, $\langle\langle q \rangle\rangle_{m_g}(t)$ と $\left\langle\left\langle q^2 \right\rangle\right\rangle_{m_g}(t)$ は

$$\langle\langle q \rangle\rangle_{m_g}(t) = (1 - \eta(t))\langle\langle q \rangle\rangle_{m_g}(t-1) + \eta(t) q(t)$$
$$\left\langle\left\langle q^2 \right\rangle\right\rangle_{m_g}(t) = (1 - \eta(t))\left\langle\left\langle q^2 \right\rangle\right\rangle_{m_g}(t-1) + \eta(t) q^2(t)$$

のように計算される.

また, この変更にともない, 上で述べたユニットの生成条件を変更する. 観測データ $\{\mathbf{S}(t), q(t)\}$ が得られたとき, 前述のユニットの生成条件 $\max_m p(\mathbf{S}(t), q(t)|m, \theta(t-1))$ が小さく, かつ

$$p(q(t)|m_g, \theta) = (2\pi)^{-\frac{1}{2}} \sigma^{-1}_{m_g} \exp\left[-\frac{(q(t) - q_{m_g})^2}{2\sigma^2_{m_g}} \right] \tag{5.22}$$

が小さい場合, 観測データ $\{\mathbf{S}(t), q(t)\}$ を説明できる通常のユニットを生成する.

B.2 平坦ユニット

通常ユニットの線形回帰行列 $\mathbf{K}_m$ は式 (5.17) によって計算される. ユニット m の入力分散が小さくなると, $\tilde{\Lambda}_m$ が発散するため, 線形回帰行列 $\mathbf{K}_m$ が推定できなくなる. そこで, 入力分散が小さなユニットの線形回帰行列 $\mathbf{K}_m$ を 0 とした.

これにともない, モデルパラメータ b_m と σ^2_m は

$$b_m = \frac{\langle\langle q \rangle\rangle_m(t)}{\langle\langle 1 \rangle\rangle_m(t)}$$
$$\sigma^2_m = \frac{\left\langle\left\langle (q - b_m(t))^2 \right\rangle\right\rangle_m(t)}{\langle\langle 1 \rangle\rangle_m(t)} = \frac{\left\langle\left\langle q^2 \right\rangle\right\rangle_m(t) - b^2_m(t)\langle\langle 1 \rangle\rangle_m(t)}{\langle\langle 1 \rangle\rangle_m(t)}$$

のように更新される. また, 教師データ $\{\mathbf{S}(t), q(t)\}$ が与えられたとき, $\langle\langle q \rangle\rangle_m(t)$ と $\left\langle\left\langle q^2 \right\rangle\right\rangle_m(t)$ は

$$\langle\langle q \rangle\rangle_m(t) = (1 - \eta(t))\langle\langle q \rangle\rangle_m(t-1) + \eta(t) q(t)$$
$$\left\langle\left\langle q^2 \right\rangle\right\rangle_m(t) = (1 - \eta(t))\left\langle\left\langle q^2 \right\rangle\right\rangle_m(t-1) + \eta(t) q^2(t)$$

と計算される.

B.3 M-step の停止

E-step の計算において $P_m(t) << 1$ となるユニットに対して, M-step の処理を行うと, 式 (5.20) の重み付き平均 $\langle\langle\cdot\rangle\rangle$ はほぼ $1-\eta(t)$ 倍されるのと同様である. 一方, 例えばモデルパラメータ μ_m は式 (5.15) で更新されるが, $\langle\langle\mathbf{S}\rangle\rangle_m(t)$ と $\langle\langle 1\rangle\rangle_m(t)$ が共に定数倍されても推定値は変化しない. $P_m(t) << 1$ となる観測データが連続して観測されると, ユニット m の保持する重み付き平均 $\langle\langle\cdot\rangle\rangle_m$ が小さくなり, 数値誤差による影響を受けやすくなる. 特に, 高次元の関数近似問題で, 強化学習問題のようにマルコフ連鎖に従って観測データが与えられる場合には $P_m(t) << 1$ となる観測データが連続して得られる可能性が高い. そこで, $P_m(t) << 1$ となるユニット m の重み付き平均とモデルパラメータの更新 (M-step) を行わない.

NGnet は局所モデルであり観測データが与えられたときに $P_m(t) >> 0$ となるユニットは少数であることが多い. よって, 多くのユニットについて M-step を行わないことになり, この修正は計算量削減の点からもメリットがある.

C. 方策パラメータに関する勾配

方策パラメータに関する勾配が

$$\frac{\partial\rho(\boldsymbol{\theta})}{\partial\theta_i} = \int_{\mathbf{s}} D(\mathbf{s})ds \int_{\mathbf{u}} d\mathbf{u}\frac{\partial\log\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial\theta_i}Q^{\pi_\theta}(\mathbf{u}|\mathbf{s}) \tag{5.23}$$

となることを示す.

$$\begin{aligned}
\frac{\partial V_{\boldsymbol{\theta}}(\mathbf{s})}{\partial\theta_i} &\equiv \frac{\partial}{\partial\theta_i}\int_{\mathbf{u}} d\mathbf{u}\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \\
&= \int_{\mathbf{u}} d\mathbf{u}\left[\frac{\partial\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial\theta_i}Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) + \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})\frac{\partial}{\partial\theta_i}Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})\right] \\
&= \int_{\mathbf{u}} d\mathbf{u}\left[\frac{\partial\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial\theta_i}Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) + \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})\frac{\partial}{\partial\theta_i}\left[r(\mathbf{s},\mathbf{u}) - \rho(\boldsymbol{\theta}) + \int_{\mathbf{s}'} d\mathbf{s}' p(\mathbf{s}'|\mathbf{s},\mathbf{u})V_{\boldsymbol{\theta}}(\mathbf{s}')\right]\right] \\
&= \int_{\mathbf{u}} d\mathbf{u}\left[\frac{\partial\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial\theta_i}Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) + \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})\left[-\frac{\partial\rho(\boldsymbol{\theta})}{\partial\theta_i} + \frac{\partial}{\partial\theta_i}\int_{\mathbf{s}'} d\mathbf{s}' p(\mathbf{s}'|\mathbf{s},\mathbf{u})V_{\boldsymbol{\theta}}(\mathbf{s}')\right]\right]
\end{aligned}$$

よって,

$$\frac{\partial\rho(\boldsymbol{\theta})}{\partial\theta_i} = \int_{\mathbf{u}} d\mathbf{u}\left[\frac{\partial\pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial\theta_i}Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) + \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})\int_{\mathbf{s}'} d\mathbf{s}' p(\mathbf{s}'|\mathbf{s},\mathbf{u})\frac{\partial V_{\boldsymbol{\theta}}(\mathbf{s}')}{\partial\theta_i}\right] - \frac{\partial V_{\boldsymbol{\theta}}(\mathbf{s})}{\partial\theta_i}$$

この両辺の状態 $\mathbf{s}$ の定常分布 $D_{\boldsymbol{\theta}}(\mathbf{s})$ に関する期待値を取ると,

$$\begin{aligned}\int_{\mathbf{s}} d\mathbf{s} D_{\boldsymbol{\theta}}(\mathbf{s}) \frac{\partial \rho(\boldsymbol{\theta})}{\partial \theta_i} &= \int_{\mathbf{s}} d\mathbf{s} D_{\boldsymbol{\theta}}(\mathbf{s}) \int_{\mathbf{u}} d\mathbf{u} \frac{\partial \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial \theta_i} Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \\ &\quad + \int_{\mathbf{s}} d\mathbf{s} \int_{\mathbf{u}} d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s}) \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s}) \int_{\mathbf{s}'} d\mathbf{s}' p(\mathbf{s}'|\mathbf{s},\mathbf{u}) \frac{\partial V_{\boldsymbol{\theta}}(\mathbf{s}')}{\partial \theta_i} \\ &\quad - \int_{\mathbf{s}} d\mathbf{s} D_{\boldsymbol{\theta}}(\mathbf{s}) \frac{\partial V_{\boldsymbol{\theta}}(\mathbf{s}')}{\partial \theta_i}\end{aligned}$$

となる. ここで, マルコフ過程の定常性から $D_{\boldsymbol{\theta}}(\mathbf{s}') = \int_{\mathbf{s}} \int_{\mathbf{u}} d\mathbf{s} d\mathbf{u} p(\mathbf{s}'|\mathbf{s},\mathbf{u}) \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s}) D_{\boldsymbol{\theta}}(\mathbf{s})$ となるため第 2 項と第 3 項は消去されて,

$$\int_{\mathbf{s}} d\mathbf{s} D_{\boldsymbol{\theta}}(\mathbf{s}) \frac{\partial \rho(\boldsymbol{\theta})}{\partial \theta_i} = \int_{\mathbf{s}} d\mathbf{s} D_{\boldsymbol{\theta}}(\mathbf{s}) \int_{\mathbf{u}} d\mathbf{u} \frac{\partial \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial \theta_i} Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u})$$

となる. また,

$$\begin{aligned}\frac{\partial \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})}{\partial \theta_i} &= \pi_{\boldsymbol{\theta}}(\mathbf{s}) \frac{\partial}{\partial \theta_i} \ln \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s}) \\ &= \pi_{\boldsymbol{\theta}}(\mathbf{s}) \psi_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \end{aligned} \tag{5.24}$$

であるから,

$$\int_{\mathbf{s}} d\mathbf{s} D_{\boldsymbol{\theta}}(\mathbf{s}) \frac{\partial \rho(\boldsymbol{\theta})}{\partial \theta_i} = \int_{\mathbf{s},\mathbf{u}} d\mathbf{s} d\mathbf{u} D_{\boldsymbol{\theta}}(\mathbf{s}) \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s}) Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \psi_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) \tag{5.25}$$

となる. また定常性から $D_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{u}) = D_{\boldsymbol{\theta}}(\mathbf{s}) \pi_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{s})$ である.

謝辞

研究の全般にわたってご指導下さった ATR 脳情報研究所の 佐藤 雅昭 先生, 指導教官として研究全般について指導して頂いた 石井 信 教授に感謝します. 私がこの研究を行うことができたのも, お二方のご指導によるものだと思います. ありがとうございました. また, 日頃より研究に関して議論してくださった 柴田 智広 助教授, 大羽 成征 助手, 吉本 潤一郎 博士に感謝します. 研究に必要な資源である計算機の管理をして頂いた作村 勇一 助手 に感謝します. 本研究で行ったシミュレーション実験の実装と実行に協力して頂いた 森 健 君, お礼申し上げます. そして, 論理生命学分野の皆様, 研究活動や日常生活の様々なところで支えて頂きました. ありがとうございました. また, 両親を初め, 家族からのサポートに対して感謝します. 最後に, 論文審査を引き受けて頂いた 石井 信 教授, 小笠原 司 教授, 杉本 謙二 教授, 柴田 智広 助教授に感謝します.

参考文献

[1] T. McGeer: “Passive dynamic walking”, International journal of Robotics Research, **9**, 2, pp. 62–82 (1990).

[2] S. Mochon and T. A. McMahon: “Ballistic walking”, Journal of Biomechanics, **13**, pp. 49–57 (1980).

[3] K. Hirai, M. Hirose, Y. Haikawa and T. Takenaka: “The development of honda humanoid robot”, Proceedings of the 1998 IEEE International Conference on Robotics & Automation (1998).

[4] R. S. Sutton and A. G. Barto: “Reinforcement Learning: An Introduction”, MIT Press (1998).

[5] A. H. Cohen: “Control principles for locomotion - looking toward biology”, 2nd International Symposium on Adaptive Motion of Animals and Machines (2003).

[6] S. Grillner, P. Wallen, L. Brodin and A. Lansner: “Neuronal network generating locomotor behavior in lamprey: circuitry, transmitters, membrane properties and simulations”, Annual Review of Neuroscience, **14**, pp. 169–199 (1991).

[7] O. Andersson and S. Grillner: “Peripheral control of the cat’s step cycle. ii. entrainment of the central pattern generators for locomotion by sinusoidal hip movements during “fictive locomotion””, Acta Physiologica Scandinavica, **118**, pp. 229–239 (1983).

[8] S. Grillner: “Locomotion in vertebrates: central mechanisms and reflex interaction”, Physiological Reviews, **55**, 2, pp. 247–304 (1975).

[9] Ö. Ekeberg: “A combined neuronal and mechanical model of fish swimming”, Biological Cybernetics, **69**, pp. 363–374 (1993).

[10] G. Taga, Y. Yamaguchi and H. Shimizu: "Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment", Biological Cybernetics, **65**, pp. 147–159 (1991).

[11] G. Taga: "A model of the neuro-musculo-skeletul system for human locomotion i", Biological Cybernetics, **73**, pp. 97–111 (1995).

[12] K. Doya and S. Yoshizawa: "Adaptive synchronization of neural and physical oscillators", Advances in Neural Information Processing Systems, **4**, pp. 109–116 (1992).

[13] A. J. Ijspeert and J.-M. Cabeluguen: "Gait transition from swimming to walking: investigation of salamander locomotion control using nonlinear oscillators" (2003).

[14] N. Ogihara and N. Yamazaki: "Generation of human bipedal locomotion by a bio-mimetic neuro-musculo-skeletal model", Biological Cybernetics, **84**, pp. 1–11 (2001).

[15] R. S. Sutton: "Learning to predict by the methods of temporal differences", Machine Learning, **3**, pp. 9–44 (1988).

[16] G. Rummery and M. Niranjan: "line q-learning using connectionist systems" (1994).

[17] A. G. Barto, R. S. Sutton and C. W. Anderson: "Neuronlike adaptive elements that can solve difficult learning control problems", IEEE Transactions on Systems, Man, and Cybernetics, 13, pp. 834–846 (1983).

[18] C. J. C. H. Watkins and P. Dayan: "Qlearning", Machine Learning, **8**, pp. 279–292 (1992).

[19] D. P. Bertsekas and J. N. Tsitsiklis: "Neuro-dynamic programming", Atheta Scientific, Belmont, MA (1996).

[20] K. Doya: "Reinforcement learning in continuous time and space", Neural Computation, **12**, pp. 243–269 (1999).

[21] J. Morimoto and K. Doya: "Robust reinforcement learning", Advances in Neural Information Processing Systems, Vol. 13, pp. 1061–1067 (2000).

[22] M. Sato and S. Ishii: "Reinforcement learning based on on-line em algorithm", Advances in Neural Information Processing Systems, **11**, pp. 1052–1058 (1999).

[23] J. Yoshimoto, S. Ishii and M. Sato: "Application of reinforcement learning to balancing of acrobot", in Proceedings of 1999 IEEE International Conference on Systems, Man and Cybernetics, V, pp. 516–521 (1999).

[24] J. Yoshimoto, S. Ishii and M. Sato: "On-line em reinforcement learning", IEEE-INNS-ENNS international Joint Conference on Neural Networks(IJCNN 2000), Vol. 3, pp. 163–168 (2000).

[25] J. Morimoto and K. Doya: "Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning", Robotics and Autonomous Systems, **36**, pp. 37–51 (2001).

[26] J. Morimoto and C. G. Atkeson: "Minimax differential dynamic programming: An application to robust biped walking", Advances in Neural Information Processing Systems, **15**, pp. 1539–1546 (2003).

[27] S. P. Singh, T. Jaakkola and M. I. Jordan: "Learning without state-estimation in partially observable markovian decision processes", Proceedings of the 11th International Conference on Machine Learning, pp. 284–292 (1994).

[28] M. Sato: "A real time learning algorithm for recurrent analog neural networks", Biological Cybernetics, **62**, pp. 237–241 (1990).

[29] R. J. Williams and D. Zipser: "Gradient-based algorithm for recurrent network and their computational complexity", Hillsdale, NJ: Erlbaum (1995).

[30] H. J. Kushner and G. G. Yin: "Stochastic Approximation Algorithms and Applications", New York: Springer-Verlag (1997).

[31] M. Sato and S. Ishii: "On-line em algorithm for the normalized gaussian network", Neural Computation, **12**, pp. 407–432 (2000).

[32] J. Moody and C. J. Darken: "Fast learning in networks of locally-tuned processing units", Neural Computation, **1**, pp. 281–294 (1989).

[33] A. P. Dempster, N. M. Laird and D. B. Rubin: "Maximum likelihood from incomplete data via the em algorithm", Journal of the Royal Statistical Society, Series B (Methodological), **39**, 1, pp. 1–38 (1977).

[34] D. A. White and D. A. Sofge: "Applied learning : Optimal control for manufacturing", pp. 259–282 (1992).

[35] V. R. Konda and J. N. Tsitsiklis: "Actor-critic algorithms", SIAM Journal on Control and Optimization, **42**, 4, pp. 1143–1146 (2003).

[36] R. J. Williams: "Simple statistical gradient-following algorithms for connectionist reinforcement learning", Machine Learning, **8**, pp. 229–256 (1992).

[37] H. Kimura and S. Kobayashi: "An analysis of actor/critic algorithms using eligibility traces: Reinforcement learning with imperfect value function", 15th International Conference on Machine Learning, pp. 278–286 (1998).

[38] R. S. Sutton, D. McAllester, S. Singh and Y. Manour: "Policy gradient method for reinforcement learning with function approximation", Proceedings of the 1998 IEEE International Conference on Robotics & Automation (2000).

[39] P. Marbach and J. N. Tsitsiklis: "Simulation-based optimization of markov reward processes", IEEE Transactions on Autumatic Control, **46**, 2, pp. 191–209 (2001).

[40] J. Tsitsiklis and B. V. Roy: "Average cost temporal-difference learning" (1997).

[41] S. Kakade: "A natural policy gradient", In Advances in Neural Information Processing Systems, **14**, pp. 1531–1538 (2001).

[42] S. Amari: "Natural gradient works efficiently in learning", Neural Computation, **10**, 2, pp. 251–276 (1998).

[43] J. Peters, S. Vijayakumar and S. Schaal: "Reinforcement learning for humanoid robotics", Third IEEE International Conference on Humanoid Robotics 2003, Germany (2003).

[44] S. J. Bradtke and A. G. Barto: "Linear least-squares algorithms for temporal difference learning", Machine Learning, **22**, pp. 33–57 (1996).

[45] J. A. Boyan: "Least-squares temporal difference learning", Proc. 16th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA, pp. 49–56 (1999).

[46] M. G. Lagoudakis, R. Parr and M. L. Littman: "Least-squares methods in reinforcement learning for control", SETN, pp. 249–260 (2002).

研究業績

論文

- 中村 泰, 佐藤 雅昭, 石井 信: 神経振動子ネットワークを用いたリズム運動に対する強化学習法, 電子情報通信学会論文誌, Vol. J87–D-II(3), pp.893–902, (2004)

国際会議

- Sato, M., Nakamura, Y., and Ishii, S. Reinforcement learning for biped locomotion. International Conference on Artificial Neural Networks (ICANN 2002), Lecture Notes in Computer Science, 2415, Springer-Verlag, pp.777-782 (2002).
- Nakamura, Y., Sato, M. and Ishii, S.: Reinforcement learning for biped robot, *2nd International Symposium on Adaptive Motion of Animals and Machines*. ThP-II-5, http://www.kimura.is.uec.ac.jp/amam2003/PAPERS/E07-nakamura.pdf, (2003)

研究会

- 中村 泰, 石井 信, 佐藤 雅昭: 神経振動子ネットワークを用いた強化学習法による歩行運動の獲得. 電子情報通信学会ニューロコンピューティング研究会 (電子情報通信学会技術研究報告, NC2001-156, 183-190). (2002).
- 中村 泰, 石井 信, 佐藤 雅昭: 強化学習による 2 足歩行の獲得. 脳と心のメカニズム第 3 回夏のワークショップ 「知能発達のメカニズム」, (2002)
- 中村 泰, 佐藤 雅昭, 石井 信. 神経振動子ネットワークを用いた強化学習法による２足歩行運動の獲得. 日本神経回路学会第 13 回全国大会, pp.74-75, (2003).

その他の業績

- 中村 泰, 大羽 成征, 吉本 潤一郎, 石井 信: オンラインベイズ法によるヒトの指さし運動の解析. 電子情報通信学会ニューロコンピューティング研究会 (電子情報通信学会技術研究報告, NC2002-158, pp.149-154), (2003).