ロボット学習

森本　淳

本稿では，特にヒューマノイドロボットのような多くのセンサ入力，多くのアクチュエータ出力を有する系において，制御出力情報とセンサ情報にもとづいて目的とする制御システムを構築するロボット学習のアプローチについての解説を行う．

キーワード：ロボット学習，最適制御，強化学習，ヒューマノイドロボット

1. はじめに

ロボットの制御は，制御理論の分野から派生し，おもに多自由度の剛体リンク系を制御する方法および車輪をもつ移動ロボットの環境認識の手法などに方向性を特化したうえで，部品組み付けなどを行う産業用ロボット制御や工場内での物品の自動搬送などの応用において実用化され，日本を代表する技術として社会に貢献している．その一方でヒューマノイドロボットや屋外での移動ロボットの自律制御など，動的に変化する複雑な環境に適応する制御技術については研究段階にあり実用レベルに到達することは容易でないと考えられてきた．その主な要因の一つに，実環境をモデルとして記述するには大量・多次元の情報を必要とし，かつそのすべての情報を計測できるわけではないことが挙げられる．ところが，近年の計算機能力の飛躍的な向上と機械学習アルゴリズムの発展を背景に，獲得される大規模データとその情報処理による，実環境に対応可能な制御システムの構築が現実的となってきた．本稿では，特にヒューマノイドロボットのような多くのセンサ入力，多くのアクチュエータ出力を有する系において，制御出力情報とセンサ情報にもとづいて目的とする制御システムを構築するロボット学習のアプローチについての解説を行う．

移動ロボットの自己位置同定やナビゲーション問題についても機械学習手法が広く活用され，米国における DARPA Urban Challenge などを通じて実用レベルの自律走行システムが開発されつつあり，それらの技術は Google などが応用する方向で開発を進めている [1]．これらの話題は本稿では扱わないこととするが，[2, 3] などのテキストや文献を参考にされたい．

もりもと　じゅん
ATR 脳情報研究所 ブレインロボットインタフェース研究室
〒 619–0288 京都府相楽郡精華町光台 2–2–2

2. ロボット学習問題

次のようなシステムに対する制御問題を考える．

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{1}$$

そこでロボットが学習アルゴリズムを通じて次のような制御則を獲得する問題を扱う．

$$\boldsymbol{u}(t) = \boldsymbol{\mu}(\boldsymbol{x}, t; \boldsymbol{w}) \tag{2}$$

ここで，$\boldsymbol{x}$ はシステムの状態変数を表し，$\boldsymbol{u}$ は制御入力を表す．制御則を表現するパラメータ $\boldsymbol{w}$ は次の目的関数を最小化するように学習される．

$$J = \Phi(T) + \int_0^T r(\boldsymbol{x}, \boldsymbol{u}, t)dt \tag{3}$$

ただし，r はコスト関数，T は終端時刻，Φ は終端コストを表す．多くの場合，制御に焦点をおいたロボット学習の問題は，目的関数という形で表現されるタスクを達成するための制御則 $\boldsymbol{\mu}$ を，ロボット自身の経験を通じていかに効率的に獲得するかを考える問題となる．このように目的関数を最小化するための制御則を求める問題は，一般には最適制御問題として知られており，ロボットの分野においては，動的計画法 [4] や強化学習法 [5] を用いてこの問題に取り組む研究がさかんに行われている．

2.1 最適制御則

ここで，式 (3) における目的関数を最小化するような制御入力を求める問題を考える．ある状態 $\boldsymbol{x}$ において時間 t から終端時刻 T までのコストの積算を価値関数として次のように定義する．

$$V(\boldsymbol{x}, t) = \Phi(T) + \int_t^T r(\boldsymbol{x}, \boldsymbol{u}, t)dt \tag{4}$$

この価値関数は，次に示すハミルトン・ヤコビ・ベルマン (HJB) 方程式を解くことにより得られる [6]．

$$-\frac{\partial V(t)}{\partial t} = \min_{\boldsymbol{u}} \left[r(\boldsymbol{x}, \boldsymbol{u}, t) + \frac{\partial V(t)}{\partial \boldsymbol{x}}^{\top} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \right] \quad (5)$$

たとえば，コスト関数が次のような形式の場合を考える．

$$r(t) = q(\boldsymbol{x}, t) + \frac{1}{2} \boldsymbol{u}^{\top} \boldsymbol{R} \boldsymbol{u} \quad (6)$$

すると，上述の HJB 方程式の右辺を最大にする制御則は次のように表される．

$$\boldsymbol{u}(t) = -\boldsymbol{R}^{-1} \frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}} \frac{\partial V(t)}{\partial \boldsymbol{x}} \quad (7)$$

制御対象が線形でありコスト関数が 2 次で与えられるときは，定常解については価値関数を代数的に導くことが可能である．一方で，非線形の制御対象に対しては，この価値関数を解析的に求めることが一般に困難である．

そこで，制御対象のモデルが既知である場合には，時間逆方向に価値関数を伝搬させる動的計画法が用いられる [4]．また，制御対象の特性が未知である場合には，価値関数の推定値を逐次的に伝搬させる Temporal difference 学習法 [7] などが，強化学習アルゴリズムの一部分として広く用いられてきた．このほかに，制御則のパラメータを直接最適化する Policy gradient 法などがある [8, 9, 10].

2.2 軌道ベースの最適制御

あらゆる状態 $\boldsymbol{x}$ に対して上述の制御則を求めることは，ヒューマノイドロボットなど多くの状態変数をもつシステムに対しては計算量が膨大になり困難となる．そこで状態空間全域で制御則を求めることをあきらめる一方で，高次元状態空間中のある軌道近傍での制御則を導出することが現実的なアプローチと考えられている [11]．具体的には，与えられた初期軌道まわりで，価値関数の 2 次近似を導出し，その近似された価値関数を最小化するような制御則を求める方法が提案されている [12].

この手法では，以下の計算により，時間逆方向に価値関数の 2 次モデルを伝搬させる．

$$\dot{V}_{\boldsymbol{x}} = -Q_{\boldsymbol{x}} + Q_{\boldsymbol{u}}^{\top} Q_{\boldsymbol{u}\boldsymbol{u}}^{-1} Q_{\boldsymbol{u}\boldsymbol{x}} \quad (8)$$

$$\dot{V}_{\boldsymbol{x}\boldsymbol{x}} = -Q_{\boldsymbol{x}\boldsymbol{x}} + Q_{\boldsymbol{x}\boldsymbol{u}} Q_{\boldsymbol{u}\boldsymbol{u}}^{-1} Q_{\boldsymbol{u}\boldsymbol{x}} \quad (9)$$

ただし，添字はその変数に対する偏微分を表す．ここで，上式の右辺に現れる Q 関数（行動価値関数と呼ばれる）は次のように計算される．

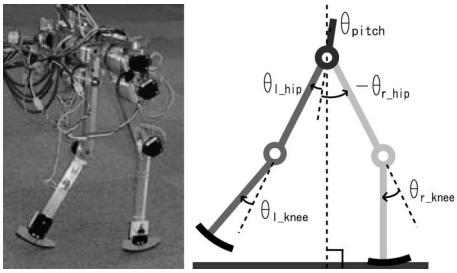

図 1 4 関節歩行ロボット．体幹の自由度を含め 5 自由度，（関節角速度を考慮し）10 次元の状態空間における制御則を学習．

$$Q_{\boldsymbol{x}} = \boldsymbol{f}_{\boldsymbol{x}} V_{\boldsymbol{x}} + r_{\boldsymbol{x}} \quad (10)$$

$$Q_{\boldsymbol{u}} = \boldsymbol{f}_{\boldsymbol{u}} V_{\boldsymbol{x}} + r_{\boldsymbol{u}} \quad (11)$$

$$Q_{\boldsymbol{x}\boldsymbol{x}} = \boldsymbol{f}_{\boldsymbol{x}\boldsymbol{x}} V_{\boldsymbol{x}} + \boldsymbol{f}_{\boldsymbol{x}} V_{\boldsymbol{x}\boldsymbol{x}} + V_{\boldsymbol{x}\boldsymbol{x}} \boldsymbol{f}_{\boldsymbol{x}}^{\top} + r_{\boldsymbol{x}\boldsymbol{x}} \quad (12)$$

$$Q_{\boldsymbol{x}\boldsymbol{u}} = \boldsymbol{f}_{\boldsymbol{x}\boldsymbol{u}} V_{\boldsymbol{x}} + V_{\boldsymbol{x}\boldsymbol{x}} \boldsymbol{f}_{\boldsymbol{u}}^{\top} + r_{\boldsymbol{x}\boldsymbol{u}} \quad (13)$$

$$Q_{\boldsymbol{u}\boldsymbol{x}} = \boldsymbol{f}_{\boldsymbol{u}\boldsymbol{x}} V_{\boldsymbol{x}} + \boldsymbol{f}_{\boldsymbol{u}} V_{\boldsymbol{x}\boldsymbol{x}} + r_{\boldsymbol{u}\boldsymbol{x}} \quad (14)$$

$$Q_{\boldsymbol{u}\boldsymbol{u}} = \boldsymbol{f}_{\boldsymbol{u}\boldsymbol{u}} V_{\boldsymbol{x}} + r_{\boldsymbol{u}\boldsymbol{u}} \quad (15)$$

ここで，$V_{\boldsymbol{x}}^i$ を $V_{\boldsymbol{x}}$ の，f^i を $\boldsymbol{f}$ のそれぞれ i 番目の要素とすると，たとえば $\boldsymbol{f}_{\boldsymbol{x}\boldsymbol{x}} V_{\boldsymbol{x}}$ は，$\boldsymbol{f}_{\boldsymbol{x}\boldsymbol{x}} V_{\boldsymbol{x}} = \sum_i f_{\boldsymbol{x}\boldsymbol{x}}^i V_{\boldsymbol{x}}^i$ として計算される．また，上式によって導出される行動価値関数を用いて，制御出力の更新量 $\delta \boldsymbol{u}$ が次のように得られる．

$$\delta \boldsymbol{u}(t) = -Q_{\boldsymbol{u}\boldsymbol{u}}^{-1} \left[Q_{\boldsymbol{u}\boldsymbol{x}} \delta \boldsymbol{x}(t) + Q_{\boldsymbol{u}} \right] \quad (16)$$

ただし，$\delta \boldsymbol{x}$ は目標軌道からのずれを表す．

上述の手法をロバスト化したものを，歩行ロボット（図 1 参照）の運動学習に適用した結果を図 2 に示した．詳細については文献 [13] を参照されたい．

3. ロボット学習の最近の展開

3.1 HJB 方程式の線形化

これまでに紹介した最適制御手法を用いて，多くのロボット学習に関する成果が得られてきた一方で，制御対象のモデルを用いないモデルフリーの学習手法は，比較的少ない自由度を持つ系への適用に限定されてきた．また，高次元システムへの適用が可能な軌道ベースの学習手法は，精度のよいモデルを必要とし，実ロボットへの応用は事例が限られていた．ところが近年，価値関数の指数関数による変換を行うと，コスト関数がある条件を満たす場合に HJB 方程式が線形化されることが指摘された [14, 15, 16]．これによって，価値関数を時間前向きに計算することが可能となった．さらに上述のアプローチを基礎として，効率的なモデ

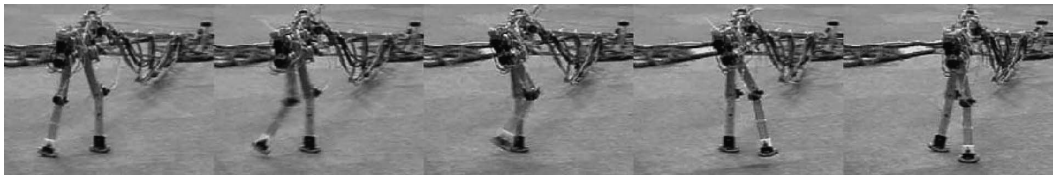

図 2　学習によって獲得された制御則によって実現されたロボットの歩行運動.

ルフリーの軌道ベース最適制御則の学習法が提案された [17]. ここでは，このロボット学習問題に対する新しいアプローチを紹介するとともに，応用事例について概説する.

まず，以下のような確率的システムを考える.

$$\dot{\boldsymbol{x}} = \boldsymbol{F}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})(\boldsymbol{u} + \boldsymbol{\epsilon}(t)) \tag{17}$$

ただし，$\boldsymbol{\epsilon}$ は分散 $\boldsymbol{\Sigma}^{\epsilon}$ のガウスノイズ入力を表す. さらに，コスト関数が次の形式を満たすとする.

$$r(t) = q(\boldsymbol{x}, t) + \frac{1}{2}\boldsymbol{u}^{\top}\boldsymbol{R}\boldsymbol{u} \tag{18}$$

すると確率システムに対する HJB 方程式は以下になる [6].

$$\begin{aligned} -V_t(\boldsymbol{x}, t) = \min_{\boldsymbol{u}} \Big[& q(\boldsymbol{x}, t) + \frac{1}{2}\boldsymbol{u}^{\top}\boldsymbol{R}\boldsymbol{u} \\ & + V_{\boldsymbol{x}}^{\top}(\boldsymbol{x}, t)(\boldsymbol{F}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}) \\ & + \frac{1}{2}Tr\big\{V_{\boldsymbol{x}\boldsymbol{x}}(\boldsymbol{x}, t)\boldsymbol{G}(\boldsymbol{x})\boldsymbol{\Sigma}^{\epsilon}\boldsymbol{G}(\boldsymbol{x})^{\top}\big\}\Big] \end{aligned} \tag{19}$$

ここで，右辺を最大化する制御則を求めると次のようになる.

$$\boldsymbol{u}(t) = -\boldsymbol{R}^{-1}\boldsymbol{G}(\boldsymbol{x})^{\top}V_{\boldsymbol{x}}(\boldsymbol{x}, t) \tag{20}$$

この制御則を式 (19) に代入すると.

$$\begin{aligned} -V_t = & \, q(\boldsymbol{x}, t) + V_{\boldsymbol{x}}(\boldsymbol{x}, t)^{\top}\boldsymbol{F}(\boldsymbol{x}) \\ & - \frac{1}{2}V_{\boldsymbol{x}}(\boldsymbol{x}, t)^{\top}\boldsymbol{G}(\boldsymbol{x})\boldsymbol{R}^{-1}\boldsymbol{G}(\boldsymbol{x})^{\top}V_{\boldsymbol{x}}(\boldsymbol{x}, t) \\ & + \frac{1}{2}Tr\{V_{\boldsymbol{x}\boldsymbol{x}}(\boldsymbol{x}, t)\boldsymbol{G}(\boldsymbol{x})\boldsymbol{\Sigma}^{\epsilon}\boldsymbol{G}(\boldsymbol{x})^{\top}\} \end{aligned} \tag{21}$$

ここで，価値関数について次の変換を導入する.

$$V(\boldsymbol{x}, t) = -\lambda \log \Psi(\boldsymbol{x}, t) \tag{22}$$

また，コスト関数について，次の拘束条件を加える.

$$\lambda\boldsymbol{R}^{-1} = \boldsymbol{\Sigma}^{\epsilon} \tag{23}$$

その結果，次の線形化された HJB 方程式が得られる.

$$\begin{aligned} -\Psi_t(\boldsymbol{x}, t) = & -\frac{1}{\lambda}\Psi(\boldsymbol{x}, t)q(\boldsymbol{x}, t) + \Psi_{\boldsymbol{x}}(\boldsymbol{x}, t)^{\top}\boldsymbol{F}(\boldsymbol{x}, t) \\ & + \frac{1}{2}Tr\left\{\Psi_{\boldsymbol{x}\boldsymbol{x}}(\boldsymbol{x}, t)\boldsymbol{G}(\boldsymbol{x})\boldsymbol{\Sigma}^{\epsilon}\boldsymbol{G}(\boldsymbol{x})^{\top}\right\} \end{aligned} \tag{24}$$

ただし，$\Psi(T) = \exp\left(-\frac{1}{\lambda}\Phi(T)\right)$ を終端条件とする. 上式は Kolmogorov の後退方程式とよばれ，Feynman-Kac の公式から次のように解が与えられる [18].

$$\Psi(\boldsymbol{x}, t) = E\left[\Psi(\boldsymbol{x}, T)\exp\left(-\int_t^T \frac{1}{\lambda}q(\boldsymbol{x}, s)ds\right)\right] \tag{25}$$

ここで $E[\cdot]$ は，軌道の出現確率についての期待値を表す. 式 (20) および (25) から，次のように最適制御則が得られる [15, 17].

$$\boldsymbol{u} = E\big[\boldsymbol{R}^{-1}\boldsymbol{G}(\boldsymbol{x})^{\top}(\boldsymbol{G}(\boldsymbol{x})\boldsymbol{R}^{-1}\boldsymbol{G}(\boldsymbol{x})^{\top})^{-1}\boldsymbol{G}(\boldsymbol{x})\boldsymbol{\epsilon}\big] \tag{26}$$

3.2　応用事例

上述の HJB 方程式の線形化の考えをもとに，モデルフリーのロボット学習アルゴリズムが提案された [17]. この学習アルゴリズムを歩行運動学習問題に適用した例を紹介する [19]. 具体的にはわれわれの研究グループで所有するヒューマノイドロボット CB-i [20, 21]（図 3a）を簡略化した歩行ロボットモデル（図 3b）を用いて，歩行学習を行った. ここでの学習アルゴリズムは軌道ベースの最適化手法であるため，もとになる周期軌道の生成手段として，正弦関数基底を出力関数としてもつ位相振動子系を用いることとした. この適用例において，多自由度・高次元の制御対象に対して制御則の学習が可能であることを示すと同時に，周期運動学習においては，時間依存の制御則ではなく振動子の位相に依存した制御則の学習を行うことが有用であることを紹介する.

周期軌道生成

ここではじめに，周期軌道の生成方法を示す. まず，歩行運動を行うロボットの位相を床反力中心点 y_{cop} とその速度 $\dot{y}_{\mathrm{cop}}$ を用いて以下のように抽出する [22].

$$\phi(\boldsymbol{y}_{\mathrm{cop}}) = -\mathrm{atan2}\left(\dot{y}_{\mathrm{cop}}, y_{\mathrm{cop}}\right) + \pi \tag{27}$$

ここで $\boldsymbol{y}_{\mathrm{cop}} = (y_{\mathrm{cop}}, \dot{y}_{\mathrm{cop}})$ である.

ここでは 10 個の振動子を制御に用い，各振動子の位相を ϕ_i $(i = 1, \ldots, 10)$ で表す. そして各振動子の位相 ϕ_i とロボットの位相 $\phi(\boldsymbol{y}_{\mathrm{cop}})$ を同期させるため，以下のように各振動子の速度を決定する.

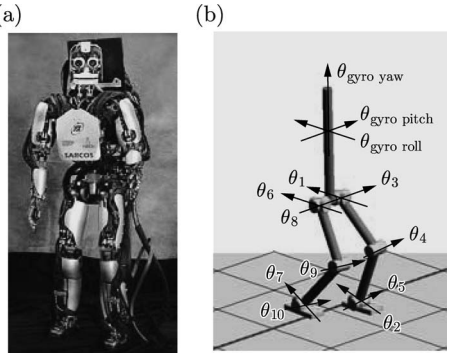

図 3 (a) ヒューマノイドロボット CB-i（身長 1.6 m，体重 90 kg，51 自由度）．(b) CB-i を簡略化した 10 関節歩行ロボットモデル．体幹の自由度を含め 13 自由度，（関節角速度を考慮し）26 次元の状態空間における制御則を学習．

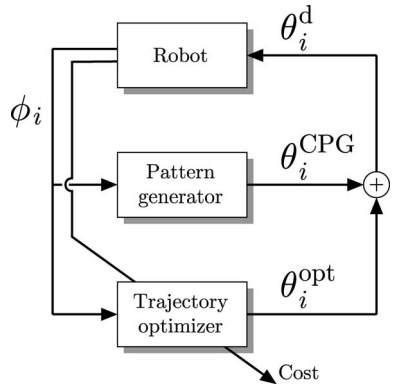

図 4 制御対象，軌道生成器，最適制御器の関係図．制御器のパラメータはコストを最小化するよう学習を通じて更新される．

$$\dot{\phi}_i = \omega + K\sin(\phi(\boldsymbol{y}_{\mathrm{cop}}) - \phi_i) \tag{28}$$

ただし ω と K は固有角周波数と結合強度を表す．

左足の腰関節と足首のロール角 (θ_1, θ_2)，そして腰と膝，足首のピッチ角 (θ_3, θ_4, θ_5) に対する目標軌道は，振動子の位相に応じた正弦波の足し合わせによって出力する [22]．同様に，右足の腰関節と足首のロール角 (θ_6, θ_7)，そして腰関節と膝，足首のピッチ角 (θ_8, θ_9, θ_{10}) に対する目標軌道も正弦波の足し合わせに従って出力されるものとした（図 3b 参照）．

各目標軌道のうち腰関節と足首のみ最適化を行い，膝関節に関しては正弦波を基底とした出力を直接用いた．

$$\theta_i^{\mathrm{d}} = \begin{cases} \theta_i^{\mathrm{CPG}} + \theta_i^{\mathrm{opt}} & (i = 1,2,3,5,6,7,8,10), \\ \theta_i^{\mathrm{CPG}} & (i = 4,9), \end{cases} \tag{29}$$

ここで θ_i^{d} は最終的に得られる目標軌道，θ_i^{CPG} は正弦波を基底とした軌道，θ_i^{opt} は学習される制御則が出力する軌道修正量を表す．この目標軌道 θ_i^{d} が Proportional Derivative (PD) 制御器に入力され，目標軌道を追従するためのトルク指令が計算される（図 4）．

歩行軌道学習

学習される制御則が出力する修正量 θ_i^{opt} は以下の式によって決定される．

$$\theta_i^{\mathrm{opt}} = \boldsymbol{g}(\phi_i)^\top (\boldsymbol{b}_i + \boldsymbol{\epsilon}(\phi_i)) \tag{30}$$

ここで $\boldsymbol{g} \in \mathcal{R}^{m\times 1}$ および $\boldsymbol{b}_i \in \mathcal{R}^{m\times 1}$，$\boldsymbol{\epsilon} \in \mathcal{R}^{m\times 1}$ はそれぞれ基底関数および学習変数，探索のためのガウスノイズである．m は位相に沿って張られる基底の個数を表す．$\boldsymbol{b}$ および $\boldsymbol{g}$ はそれぞれ式 (17) における制御量 $\boldsymbol{u}$ と入力ゲイン $\boldsymbol{G}$ に対応する．PD 制御器は各関節に実装されており，目標軌道 ($\theta_i^{\mathrm{d}} = \theta_i^{\mathrm{CPG}} + \theta_i^{\mathrm{opt}}$) を追従するためのトルク指令を計算する．

サンプルされた軌道（rollout と呼ばれる）群を用いて，式 (30) の制御則のパラメータ $\boldsymbol{b}$ は逐次的に更新される．詳細なパラメータ更新手法は [17] を参照されたい．

シミュレーション結果

ここでは，歩行ロボットモデル（図 3b）を用いたシミュレーション結果を紹介する．式 (28) における位相振動子系の固有角周波数 ω と結合係数 K はそれぞれ π rad/sec，10 とした．コスト関数は以下のとおりとした．

$$r(\phi) = k_1\left(\theta_{\text{gyro roll}}^2 + \theta_{\text{gyro pitch}}^2\right) + k_2\theta_{\text{gyro yaw}}^2 + \frac{1}{2}\sum_{i=1}^{m}\boldsymbol{b}_i^\top \boldsymbol{R}\boldsymbol{b}_i \tag{31}$$

ただし $\theta_{\text{gyro roll}}$ および $\theta_{\text{gyro pitch}}$，$\theta_{\text{gyro yaw}}$ はそれぞれロボットの上半身の傾き角度であり（図 3b 参照），$k_1 = 10^6$, $k_2 = 10^2$, $R = 0.1I$ はそれぞれコスト関数の係数である．上記のコスト関数により，上半身を鉛直に維持しながらまっすぐ歩くように制御則が学習される．

学習曲線を図 5a に表示した．縦軸と横軸はそれぞれ累積コストとパラメータの更新回数を表す．10 回の試行と 10 回のパラメータ更新における平均値と標準偏差をプロットした．位相依存の rollout を用いる提案手法と時間依存の rollout を用いる既存手法の比較を行った．実線と破線はそれぞれ位相依存と時間依存の結果を表す．いずれの方法でも目的とする歩行運動が獲得された．また，位相依存における学習曲線が時間依存に

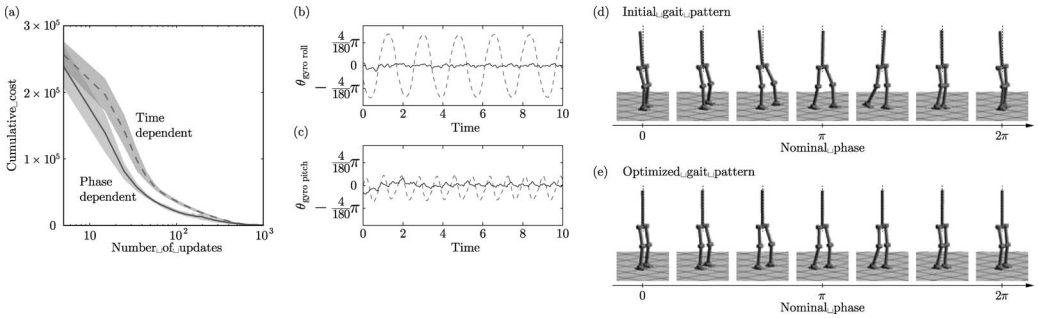

図 5 学習結果 (a) 累積コストの学習曲線．横軸，縦軸はそれぞれパラメータ更新の回数，累積コストを表す．10 回の実験と 10 回の更新における平均値と標準偏差を表示した．実線と破線はそれぞれ位相依存の手法と時間依存の手法における結果を表す．(b)(c) ロボットの上半身のロール角およびピッチ角を表している．実線と破線はそれぞれ学習後と学習前の軌道を表している．(d)(e) 学習前後における歩行パターン．縦の破線は鉛直方向を表している．

比べてはやく低下していることがわかる．図 5b および c はロボットの上半身の傾き軌道 ($\theta_{\mathrm{gyro\,roll}}$，$\theta_{\mathrm{gyro\,pitch}}$) を表示している．実線と破線はそれぞれ学習後と学習前の軌道を表している．学習前におけるロール角・ピッチ角ともに振幅が大きい（破線）．学習の結果，両者はゼロに近い値を維持している（実線）．

図 5d および e に，学習前後における歩行パターンを表示した．学習前は上半身が大きく前後左右に振れていることがわかる．一方，学習後の上半身はほぼ鉛直を維持した歩行パターンになっている．

4. おわりに

本稿では，多自由度のロボットが運動学習を行う枠組みを中心に紹介した．近年の HJB 方程式の線形化手法の導出を背景に，高次元の状態空間をもつ多自由度ロボットの運動学習が現実的となってきた．応用として，歩行運動学習を行った事例を紹介した．周期的な運動課題に対して，位相依存での表現を用いることにより学習性能が向上することを実験的に示した．現在，我々の研究グループでは，ヒューマノイドロボット CB-i（図 3a）への実装を目指して学習アルゴリズムの改良を進めている．

本稿では，制御則の最適化に焦点を当てた紹介を行ったが，他にも，見まね学習など多自由度システムにおいて効率的に運動学習を行うアプローチが多く提案されている．それらロボット学習の研究事例を集めた論文特集号として [23, 24, 25] を紹介しておく．

多くの自由度を有するシステムのための学習メカニズムを開発することは，同様に多自由度システムであるヒトが日々直面している学習問題を理解することにも通じる可能性がある．たとえば，新学術領域研究「予測と意思決定の脳内計算機構の解明による人間理解と応用」(http://www.decisions.jp) では，神経科学，分子生物学，精神医学，哲学，心理学，機械学習，ロボティクスなどの分野の研究グループが協力し人間の学習システムの理解を目指した取り組みを行っている．今後，工学応用のみならず，ロボット学習の知見が人間理解にも貢献することが期待される．

謝辞 応用事例（3.2 節）については（独）情報通信研究機構の杉本徳和氏との共同研究の成果である．本研究の一部は，科研費新学術領域研究「予測と意思決定」(23120004) の助成を受けたものである．また，文科省脳科学研究戦略推進プログラムにより実施された成果である．

参考文献

[1] J. Markoff. Google cars drive themseleves, in traffic. *The New York Times*, Oct. 9, 2010.
[2] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005.
[3] S. Thrun and M. Montemerlo. The GraphSLAM algorithm with applications to large-Scale mapping of urban structures. *The International Journal of Robotics Research*, **25**(5–6): 403–429, 2006.
[4] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2005.
[5] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, 1998.
[6] P. Dorato, C. Abdallah, and V. Cerone. *Linear-Quadratic Control: An Introduction*. Krieger Pub. Co., 2000.
[7] R. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, **3**(1): 135–170, 1988.
[8] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems 12*, 1057–1063, Cambridge, MA, 2000. MIT Press.

[9] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, **21**: 682–697, 2008.
[10] T. Matsubara, J. Morimoto, J. Nakanishi, M. Sato, and K. Doya. Learning CPG-based biped locomotion with a policy gradient method. *Robotics and Autonomous Systems*, **54**(11): 911–920, 2006.
[11] C. G. Atkeson and J. Morimoto. Nonparametric representation of policies and value functions: A trajectory-based approach. *Advances in Neural Information Processing Systems 15*, 1643–1650. MIT Press, Cambridge, MA, 2003.
[12] D. H. Jacobson and D. Q. Mayne. *Differential Dynamic Programming*. Elsevier, New York, NY, 1970.
[13] J. Morimoto and C. G. Atkeson. Minimax differential dynamic programming: An application to robust biped walking. *Advances in Neural Information Processing Systems 15*, 1563–1570. MIT Press, Cambridge, MA, 2003.
[14] W. H. Fleming and S. K. Mitter. Öptimal control and nonlinear filtering for nondegenerate diffusion processes. *Stochastics*, **8**: 63–77, 1982.
[15] H. J. Kappen. Linear theory for control of nonlinear stochastic systems. *Physical Review Letters*, **95**(20): 200201–1–200201–4, 2005.
[16] E. Todorov. Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences*, **106**(28): 11478–11483, 2009.
[17] E. Theodorou, J. Buchli, and S. Schaal. A generalized path integral control approach to reinforcement learning. *The Journal of Machine Learning Research*, **11**(Nov): 3137–3181, 2010.
[18] B. Øksendal. *Stochastic Differential Equations: An Introduction with Applications* (*6th ed.*). Springer, 2010.
[19] N. Sugimoto and J. Morimoto. Phase-dependent trajectory optimization for CPG-based biped walking using pah integral reinforcement learning. In *Proceedings of IEEE-RAS International Conference on Humanoid Robots*, 255–260, Bled, Slovenia, 2011.
[20] G. Cheng, S. Hyon, J. Morimoto, A. Ude, J. G. Hale, G. Colvin, W. Scroggin, and S. C. Jacobsen. CB: A Humanoid Research Platform for Exploring Neuroscience, *Advanced Robotics*, **21**: 1097–1114, 2007.
[21] JST-ICORP「計算脳プロジェクト」, http://www.cns.atr.jp/icorp/
[22] J. Morimoto, G. Endo, J. Nakanishi, and G. Cheng. A biologically inspired biped locomotion strategy for humanoid robots: Modulation of sinusoidal patterns by a coupled oscillator model. *IEEE Transaction on Robotics*, **24**(1): 185–191, 2007.
[23] J. Peters and A. Y. Ng. Guest editorial: Special issue on robot learning. *Autonomous Robots*, **27**(1,2): 1–2, 2009.
[24] J. Morimoto, O. C. Jenkins, and M. Toussaint. From the guest editors: Robot learning in practice. *IEEE Robotics and Automation Magazine*, **17**(2): 17–18, 2010.
[25] 森本　淳.「ロボットの運動学習」特集について. 日本ロボット学会誌, **22**(2): 1, 2004.

