強化学習とリカレントネットを用いた
並列で柔軟な学習制御システムの枠組み

大分大学　○高津聡志　後藤健太　柴田克成

A Framework of Parallel and Flexible Learning Control System Using Reinforcement Learning and a Recurrent Neural Network

Satoshi Takatsu, Kenta Goto and Katsunari Shibata　Oita University

Abstract: The authors think that constructing the whole process from sensors to motors by a neural network and learning it by reinforcement learning enable to realize autonomous, purposive, harmonious and parallel function emergence. Recently, some flexible learning systems using a neural network and reinforcement learning have been proposed. However, the learning module does not usually process the whole, but there exists some inflexible module other than the flexible learning module. In this paper, a recurrent neural network connects sensors and motors directly, and is trained by reinforcement learning. The authors claim that the system has the ability to learn the control considering various factors flexibly and in parallel according to necessity. In a simulated thrown-up-ball catching task, it is shown that adaptability to a constant external force and compensation of random external forces together with the generation of context-considered behaviors due to the use of a recurrent network emerge through learning.

１，まえがき

1.1 強化学習とニューラルネットの組み合わせ

近年、報酬や罰などの少ない情報をもとに、報酬を得て、罰を避ける合目的的な行動を、試行錯誤を通して自律的に学習する強化学習が注目を集めている。強化学習は、ロボットの行動学習でもその有効性が示されている[1][2]が、行動生成の学習に重点が置かれ、認識などの行動以外の機能とは切り離されて使用される傾向にあった。

しかし、ロボット内部で行われる処理の目的を改めて考えると、センサからの信号に基づいて、状況に応じた適切な信号をモータ(アクチュエータ)へ出力すること、つまり、センサからモータへのプロセスを何らかの評価基準で最適化することと考えられる。このような考え方に立つと、より報酬を得て、罰を避けるという明確な評価基準で学習する強化学習を用いれば、センサからモータまでのあらゆるプロセスの自律的、合目的的、かつ調和的な最適化が可能ではないかと著者らは考えている[3]。また、われわれ人間は、脳という超並列で柔軟な学習システムを持つ。今まで多くの研究が柔軟な認識、制御を目指したにも関わらず、人間のように、フレーム問題[4]に惑わされず、さまざまなことを並列に考慮し、かつ、瞬時に判断できる能力がなかなか実現できないのは、このシステムアーキテクチャの差も大きいのではないかと考えている。そして実際に、ロボットのセンサからモータまでをニューラルネットで構成し、6 千個のカラー視覚センサ信号を直接ニューラルネットに入れ、単にある物体への到達に対して報酬を与えて学習させるだけで、ロボットがさまざまな見え方の下で明るさや背景によらない物体の認識をある程度獲得することを示した[5]。

1.2 柔軟な制御の学習

柔軟な制御の学習としては、ニューラルネットがフィードフォワード制御を学習するフィードバック誤差学習が挙げられる[6]。また、過去の状態や制御入力を入力に追加することで、フィードバック制御も学習できる[7]。近年は、強化学習を利用したものも提案され、acrobot や歩行などの制御の学習で有効性が示されている[8][9]。しかしながらこれらは、与えられた目標軌道への追従を目的としたり、逆に目標軌道を学習システムの出力にして外部に別のフィードバック制御モジュールを置いたりしており、さまざまな状況を並列に考慮し、フィードバックも含めた柔軟な制御を獲得することが目的ではない。

制御における並列性という点では、Brooks の Subsumption architecture[10]による柔軟な制御が有名であり、並列性が俊敏性、柔軟性を生み、フレーム問題の解決にも有効であることが示された。しかしながら、並列に配置されたモジュール間の関係は設計者にゆだねられている。モジュールが増えれば、これは難しい問題となるであろうし、逆に人間が与えてしまうと、柔軟性を阻害する恐れがある。したがって、人間のような柔軟な制御システムを構築するためには、やはり並列システム全体を学習によって獲得していくことが必要であると考えている。過去に著者の一部は、2 関節マニュピレータのリーチング運動の学習において、hand-eye coordination、フィードフォワード制御、そして、フィードバック制御が学習できることを示唆している[11]。

そこで本研究では、リカレントニューラルネットと強化学習を用い、飛んでくるボールが落ちたところの近くに行くとしばらく報酬がもらえ、さらに外力がかかる環境でタスクの学習を行った。こうすることで、落ちてくるボールの位置予測をしながら、そこへ向かう運動の生成と、外力の影響を小さくする制御が求められる。そして、リカレントネットを用いることにより、文脈を考慮した行動計画ができるようになると同時に、一定の外力に対する適応能力、ランダムな外力に対する補償能力も学習によって獲得されることを示す。

なお、ここでは目標軌道は陽には与えられないので、予期せぬ変化に対する短期的な行動の変化をフィードバック制御と呼び、学習による長期的な適応をフィードフォワード制御と呼ぶ。

２．学習方法

ロボットは、センサ信号をリカレントニューラルネットに入力し、動作信号をモータに出力する。使用するリカレントニューラルネットは、出力ニューロンを含めたすべてのニューロンが入力信号と出力ニューロンを除く他のニューロンからの信号を入力し、その重み付け総和を 0 から 1 の値域を持つシグモイド関数を通して出力する。また、信号伝達に、1 ステップ(単位時間)の時間が経過するとした。ニューラルネットには、毎ステップ強化学習に基づいて計算された教師信号が与えられ、BPTT(Back Propagation Through Time)によって時間をさかのぼって学習される。強化学習として、連続動作に対応している actor-critic[12]を用いた。出力ニューロンは 2 個とし、一つは critic、もう一つは actor とした。また、actor の出力 a_t から 0.5 を引いて 0 対称になるようにし、そこに確率的な行動生成のための乱数 rnd_t を加え、それを定数倍してモータへの動作信号とした。critic の出力に対しては、現在の critic の出力 V_t とそのとき得られた報酬 r_t を使って、一つ前の時刻の critic の値に対する教師信号 $V_{s,t-1}$ を次式に従って生成する。

$$V_{s,t-1}=V(\vec{x}_{t-1})+\hat{r}_{t-1}=r_t+\gamma V(\vec{x}_t) \qquad (1)$$

ただし、

$$\hat{r}_{t-1}=r_t+\gamma V\left(\vec{x}_t\right)-V\left(\vec{x}_{t-1}\right) \qquad (2)$$

は TD 誤差であり、γ は割引率である。前の時刻の actor の出力 $a\left(\vec{x}_{t-1}\right)$ に対する教師信号 $a_{s,t-1}$ は、TD 誤差 $\hat{r}_{t-1}$ と、そのとき actor の出力に加えた乱数ベクトル rnd_{t-1} から

$$a_{s,t-1}=a\left(\vec{x}_{t-1}\right)+\alpha\hat{r}_{t-1}rnd_{t-1} \qquad (3)$$

と計算する。ただし、α は定数である。

また、ここでは、連続的に報酬が得られるタスクとしたので、ニューラルネットで学習することを考慮して、無限に最大報酬 $r_{\max}$ をもらい続けたときの critic の理想値が 0.9 になるように、最大報酬を

$$r_{\max}=0.9\left(1-\gamma\right) \qquad (4)$$

と決めた。今回は、γ を 0.9 としたため、最大報酬は r_{max} は 0.09 と設定した。

３．問題設定

本研究では、Fig.1 のように、玉は初期位置から斜め上方に打ち上げられ、横方向の初速度にしたがって、8~16m 進んだ図中の黒い場所の範囲でランダムに落下する。上方向の初速度は 29.4m/s で常に一定とし、玉は打ち上げ後 6 秒後に落下してその場にとどまり、15 秒後に元の位置に戻されて、新しくランダムな横方向の初速度で発射される。移動ロボットの寸法は Fig.2 の通りで、重さは 500g とした。トルクは-1~1Nm で、すべてのタイヤに等しくかけられる四輪駆動として、角速度×0.1Nm を摩擦等として減じた。また、ロボットは玉をすり抜けるようにし、衝突しないようにした。

シミュレーションの時間間隔は 0.01 秒とし、移動ロボットにとっての 1 ステップは 0.5 秒とした。また、このシミュレーション環境は、オープンソースの物理エンジンである OpenDynamicsEngine を用いて作成した。

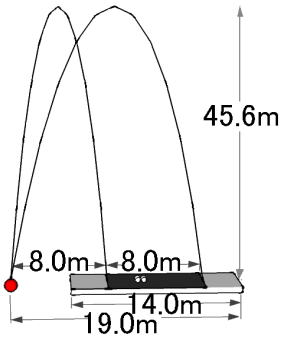

Fig.1 Environment in simulation

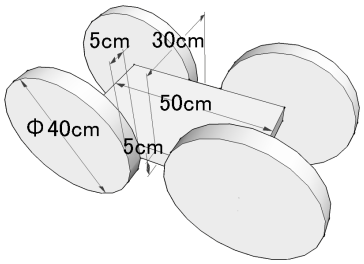

Fig.2 A moving robot in the simulation

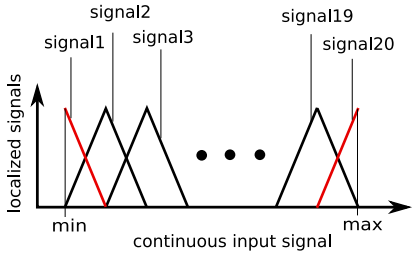

Fig.3 Localization of a continous input signals

ロボットは自分から見た玉の相対的な横方向の位置-19~14 と高さ方向の位置 0~46 を知ることができる 2 つのセンサを有しており、そこからの信号をニューラルネットへ入力する。ただし、非線形性の強い関数近似を可能とするため、1 つの連続値信号を Fig.3 のように局所化した 20 個の信号で表現してから入力した。

車輪のトルクは、actor の出力である動作信号と試行錯誤の成分である乱数を足した値を元に、ニューラルネットの 0.1~0.9 の出力が-1~1Nm になるように線形変換して用いた。その乱数は、式(5)にしたがって、最初は 2 で、100 万回後には 0.01 になるように小さくしていった。

$$\exp(\ln(0.01/2)/1000000*step)*2 \qquad (5)$$

また隠れニューロン数は、閾値として使う常に 1 を出力するニューロンを含めて 20 個とした。報酬 r は目標に近づくほど大きくするため、玉の落下点を中心にガウス関数にしたがって次式のように設定した。

$$r=r_{\max}\exp(-(x-\text{target})^2/(2\times\sigma^2)) \qquad (6)$$

なお、σ^2 は分散であり、今回は 2 とした。

４．シミュレーション

4.1 階層型ニューラルネットとの比較

リカレントネットを用いることで、文脈を用いて状態を識別し、それに基づいた制御ができるはずである。これを検証するために、リカレントネットと隠れニューロン同士の結合をなくした階層型ニューラルネットでそれぞれ学習したものを比較した。

リカレントネットでの学習後のロボットの軌跡とactor、criticの変化の値をFig.4,5に、階層型ニューラルネットでの学習後の場合をFig.6,7に示す。図のxが12mで玉が発射されており、玉の軌跡で、xの変化がなくなったところが着地点である。

Fig.4を見ると、ロボットは玉が着地する際にその位置まで進んでいることがわかる。また、リカレントネットを用いた場合のみ、玉を発射した直後は領域の中心位置に移動しようとしていた。これは中心に近いところに行った方が、落下地点の変化に対応しやすいためと考えることができる。玉が打ち上がっているのか、下がっているのかは、玉の高さの変化を捉えなければならないが、階層型では瞬間の高さしか入力されないため、その区別がつかない。そのため発射時に玉から離れるよう動く必要があっても、単に玉の方向へ近づこうとする傾向にあり、Fig.6の最後のように-4mから4mへ移動する際のように、着地に間に合わない場合もあった。Fig.6でのロボットの軌跡が曲っているのは、単に玉に近づいていく→玉が頂上にきたところで、行きすぎることに気付く→玉に合わせてactorを出力する、という動作をしているためだと考えられる。以上のことから、リカレントニューラルネットは文脈を考慮した行動を学習したと言える。

4.2 外力を付加した場合

次に、外力として、モータへのトルクに一定の大きさのトルクを加えた。外力として+0.5のトルクを加算し続けて学習した場合と、外力なしで学習した場合のそれぞれについて、テスト時に外力を加えた場合と外力を加えなかった場合の結果をFig.8～11に示す。Fig.8、11は、学習環境とテスト環境は同じであるが、Fig.9、10はテスト環境は未学習である。ここでは、4mの位置にロボットを置いた状態から、-4m、-2m、0m、2m、4mのそれぞれに玉が着地する場合のボットの軌跡を示している。

Fig.8ではロボットが動いて玉の着地の際には既に着地点に到着している。しかし、外力を加えたFig.9を見てみると、外力の方向に引っ張られて、-4mの着地点には到達できていないことがわかる。しかしFig.11を見ると、着地の瞬間には届かないながらも、玉の落下後に着地点に向かっていることがわかり、学習すれば、外力を加味した上での行動が獲得できていることがわかる。また、Fig.9ではFig.8の留まっている場所からわずかではあるが、外力の方向にずれていることがわかる。なお、玉の打ち出し初期に、既に玉の着地点にいた場合でも移動してしまっているのは、リカレントネットを用いた場合はいったん0の位置に戻ろうとするためである。

Fig.10を見てみると、Fig.11に比べて若干行きすぎてから戻っているために、着地点に間に合っていたパターンでも遅れてしまっていることがわかる。また、Fig.10は、Fig.11の留まる場所からわずかではあるが、外力がない方向にずれていることがわかる。

以上から、環境のダイナミクスが学習していないものに変化すると軌道がずれることから、外力をかけないで学習したものは外力のない状態に対応して、外力をかけて学習したものは外力のある状態に対応して、それぞれ逆ダイナミクスを、つまりフィードフォワード制御を学習していると言える。

次に、リカレントネットを用いたフィードバック制御を検証するために、毎試行-0.5～0.5のランダムな外力をトルクとして与えて学習した。そして学習後、玉が着地した後に、途中で外力を切り変えた。結果をFig.12とFig.13に示す。両グラフを見ると、それぞれ、その場に留まり続けるよう外力に対してトルクを切り替えていることが推察できる。階層型ニューラルネットで学習させた結果のFig.13では、リカレントネットと似たような出力は出るものの、リカレントネットの方が玉の位置に近くにいることから、フィードバックをうまく用いているのではないかと考えられる。

5. おわりに

センサとモータの間をリカレントニューラルネットで構成し、単に強化学習で学習することで、文脈を考慮した行動生成、一定の外力を与えた場合のフィードフォワード制御、さらには、ランダムな外力に対してはフィードバック制御の機能を、設計者の負荷なく柔軟に組み合わせた制御を行うことができることを示した。ただし、ランダムな外力を与えた場合には小さな定常誤差が残っており、本システムでは適切な積分制御を実現できなかったと考えられ、今後の課題である。

参考文献

[1] M. Asada, et al.: Purposive Behavior Acquisition for a Real Robot by Vision-Based Reinforcement Learning. Machine Learning, Vol.23, pp.279-303 (1996)

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

[3] 柴田克成: 強化学習とニューラルネットによる知能創発, 計測と制御, Vol. 48, No. 1, pp. 106-111, 2009

[4] D. Dennett: Cognitive Wheels: The Frame Problem of AI, The Philosophy of Artificial Intelligence, Margaret A. Boden, Oxford University Press, 147/170 (1984)

[5] K. Shibata and T. Kawano: Acquisition of Flexible Image Recognition by Coupling of Reinforcement Learning and a Neural Network, JCMSI, Vol. 2, No. 2, pp. 122–129 (2009)

[6] M. Kawato: Feedback-Error-Learning Neural Network for Supervised Motor Learning. Advanced neural computers (1990)

[7] 柴田克成，稲葉雅幸，井上博允: ニューラルネットによるロボットの運動学習，第6回日本ロボット学会学術講演会予稿集，pp141-142 (1988)

[8] J.Yoshimoto, et al.: Acrobot control by learning the switching of multiple controllers. Journal of AROB, 9(2), pp.67-71 (2005)

[9] T.Matsubara, et al.: Learning CPG-based Biped Locomotion with a Policy Gradient Method Robotics and Autonomus Systems, vol. 54, issue 11, pp. 911-920 (2006)

[10] R. A. Brooks: Intelligence Without Representation. Artificial Intelligence, **47**, 139/159 (1991)

[11] 柴田克成, 杉坂政典, 伊藤宏司: 強化学習によるリーチング運動の獲得,信学技報, NC2000-170, pp. 107-114 (2001)

[12] A.G.Barto: Neuronlike adaptive elements that can solve diffcult learning control problems, IEEE Trans. SMC-13, pp.834-846 (1983)

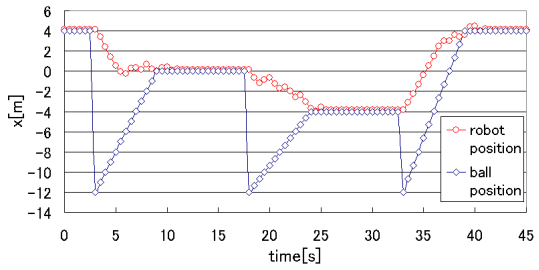

Fig.4 The locus of the robot when a recurrent neural network is used

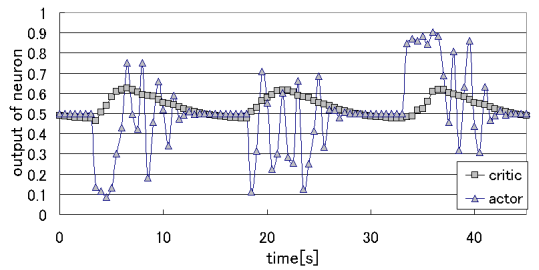

Fig.5 The change of actor and critic values when a recurrent neural network is used

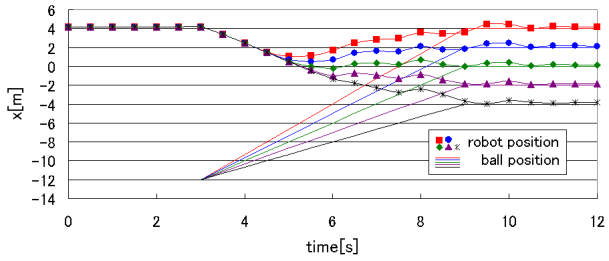

Fig.8 The loci of the robot with no external force after learning with no external force

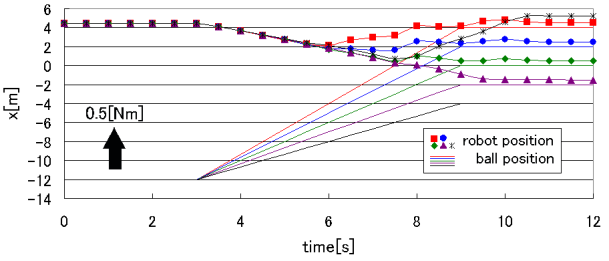

Fig.9 The loci of the robot with an external force after learning with no external force

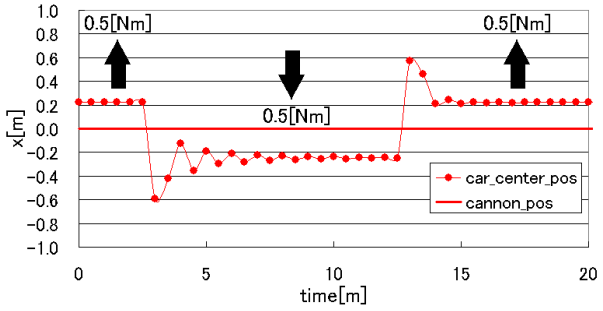

Fig.12 The locus of the robot with using a recurrent neural network when the external force changed

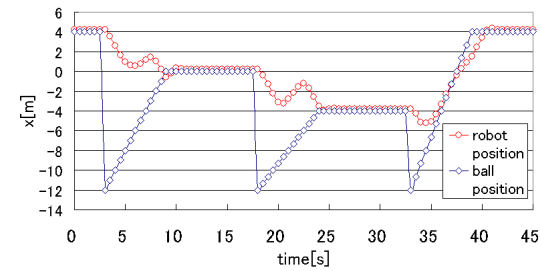

Fig.6 The locus of the robot when a layered neural network is used

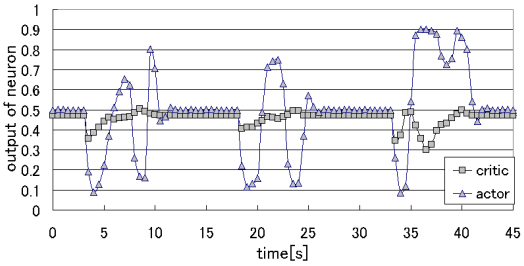

Fig.7 The change of actor and critic values when a layered neural network is used

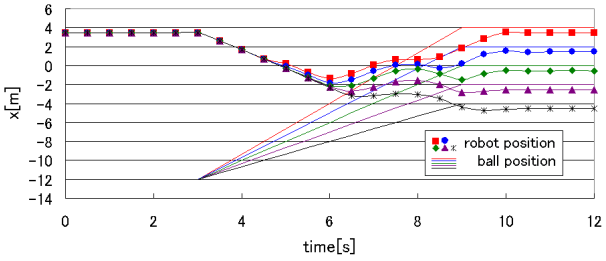

Fig.10 The loci of the robot with no external force after learning with an external force

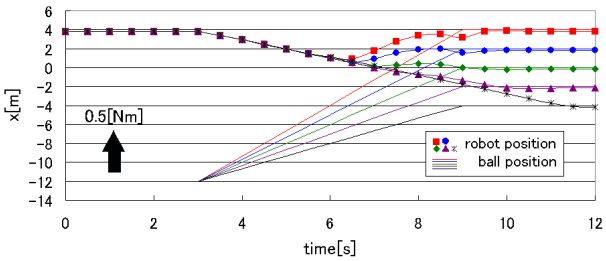

Fig.11 The loci of the robot with an external force after learning with an external force

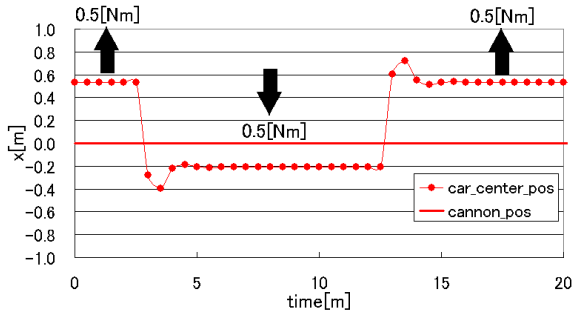

Fig.13 The locus of the robot with a layered neural network when the external force changed