

深層強化学習と言葉による離散化を用いたロボット制御への取り組み

An Approach to Robot Control using Deep Reinforcement Learning and Discretization by Words

橋本 さゆり^{*1} 金子 晃^{*2} 小林 一郎^{*3}
 Sayuri Hashimoto Akira Kaneko Ichiro Kobayashi

^{*1}お茶の水女子大学人間文化創成科学研究科理学専攻情報科学コース
 Ochanomizu University

^{*2*}お茶の水女子大学基幹研究院自然科学系
 Ochanomizu University

In recent years, the necessity for robots working in society has been spreading as the aging society has come. To easily be able to communicate with robots, it is expected that they can understand natural language and learn how to behave spontaneously through the interaction with humans. In this study, we aim to ground the meaning of natural language onto their behaviors by using reinforcement learning. In particular, we have proposed an efficient method to learn robot's motion with deep reinforcement learning by discretizing a robot's motion into a hierarchical structure consisting of basic motion elements which can be represented by words.

1. はじめに

近年、ロボットの普及が広がりつつある。今後ロボットが家庭に入ってくることが予想される。その際に人が行なった動作を見てロボットが自発的に学習することや、その学習した動作の意味を理解して人とコミュニケーションをとることが考えられる。そこで本研究では、ロボットに動作を学習させ、その学習した動作と言葉のグラウンディングを取ることでロボットに動作の意味を理解させることを試みる。具体的には、物理シミュレータ MuJoCo の上で動くロボットアーム (以下簡単のためロボットと略称する) を用い、以下の図 1 のように右側の白い円柱とタブがある状態から、左側の状態にするためにはロボットはどのような動作を行なわなければならないかを学習させる。

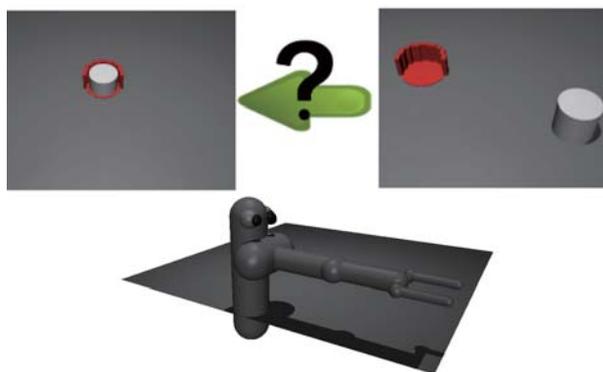


図 1: 動作の概念

本研究では、ある動作はいくつかの動作が組み合わさってできていると考える。例えば、この具体例のようにロボットが白い円柱を赤いタブにはめるためにはまずロボットが腕を移動させ、白い円柱を掴み、次に掴んだ円柱を運んで、押すという 4

連絡先: 橋本さゆり, お茶の水女子大学大学院人間文化創成科学研究科理学専攻情報科学コース小林研究室, 〒112-8610 東京都文京区大塚 2-1-1, g1320530@is.ocha.ac.jp

つの動作の構成要素に分解される。さらに掴む、運ぶ、押すという動作は下腕を上げる、下げる、指を開く、閉じるといったロボットにとっての基本的な動作の単位から成ると考える。

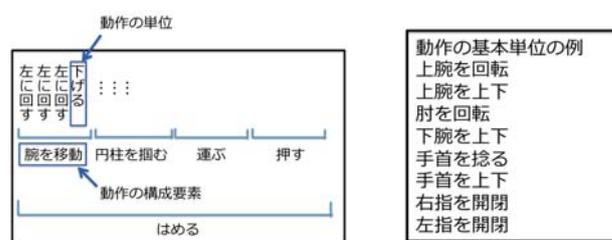


図 2: 動作の構成

2. 深層強化学習

ロボットに動作を獲得させるには強化学習という技術が用いられる。この技術は探索と評価を繰り返して行なうことで最適な行動を学習するものである。全体の枠組みとしては、学習を行なうエージェント (本研究ではロボット) がまず環境から状態を観測し、その観測結果を元に行動を行ない、その行動に対する報酬を貰うことで評価を行なう。エージェントは累積報酬の最大化を目指し、行動を行なっていく。一方で深層学習は、高次元データを扱い、高精度な結果を出すことができる。深層強化学習はこの二つの技術を組み合わせたもので、強化学習のアルゴリズムに深層学習による関数近似を組み込むことで強化学習における高次元で複雑な問題に対しても対処できるようになっている。以下に代表的な 3 つの深層強化学習の手法について説明する。

2.1 Deep Q-Network

Deep Q-Network (DQN) [2] は Atari のゲームを学習させて様々な Atari のゲームで人間以上の高得点を出したことで有名になった。DQN の元となっているのは強化学習における Q 学習というものである。Q 学習とは各状態において、可能な行動の中で最も行動価値関数の値が高い行動をとるように学習

を行なう方法であり、行動価値関数の最大値で行動を決定していくアルゴリズムである。元来 Q 学習では Q-table を用いるのが一般的であるが、Q-table では高次元データに対して次元の呪いを起こしてしまうなどの問題点があった。DQN ではこの Q-table の代わりに状態を入力とし、それぞれの行動に対する Q 値を出力とするネットワークを生成し関数近似を行なっている。

2.2 DDPG

DQN では、行動が離散的な場合しか扱うことができなかった。Deep Deterministic Policy Gradients (DDPG) [3] は行動が連続なものに対して使うことができる。DDPG の元となっているのは強化学習における Actor-critic という手法である。Actor-critic とは価値関数とは独立に、方策を表現する構造を別を持っており、Actor に基づき行動選択、Critic が報酬を得て誤差を計算するというアルゴリズムである。Actor-critic は特にロボットなどの強化学習に使用されることが多い。これらを深層化したものが DDPG である。

2.3 A3C

Asynchronous Advantage Actor-critic method (A3C) では、DDPG と同様に強化学習における Actor-critic という手法が元になっている。A3C で最も特徴的なのは GPU を用いず、CPU のスレッドを使って複数のエージェントを学習させることである。それぞれのスレッドで別々のエージェントが学習を行ない勾配更新を非同期に行なうことで、より少ない時間で高性能な学習をすることができるという利点を持つ。

我々は最初に DQN を用いて実験を開始し、ついで DDPG を用いてみた。しかし、DQN はパラメータ調整が難しく学習が困難であった。また DDPG は行動が連続値の場合にしか使用できず、学習に多大な時間がかかることから我々のロボットシミュレータが動く PC 環境では学習を完成させるには至らなかった。よって本研究では最終的に A3C を用いることとした。

3. 動きの辞書による動作の離散化

本研究では実験の際に動作を以下のような辞書にして学習を行なっている。

| 名称 | 方向 | ID | 方向 | ID | 動きの単位 |
|-------|----|----|----|----|------------|
| 上腕を回転 | 左 | A | 右 | B | 0.05 × 60 |
| 上腕を上下 | 下 | C | 上 | D | 0.05 × 60 |
| 肘を回転 | 右 | E | 左 | F | 0.05 × 60 |
| 下腕を上下 | 下 | G | 上 | H | 0.005 × 60 |
| 手首を捻る | 右 | I | 左 | J | 0.05 × 60 |
| 手首を上下 | 下 | K | 上 | L | 0.05 × 60 |
| 右指を開閉 | 左 | M | 右 | N | 0.05 × 60 |
| 左指を開閉 | 左 | O | 右 | P | 0.05 × 60 |

この表にある動きの単位とは、ロボットの各関節の最小単位の動作 0.05 を 60 回繰り返したものを指している。これによってロボットの連続な動作を離散化し

- 1 学習時間の短縮
- 2 言語との対応付けの容易化

という効用を期待した。

4. 実験

動作を離散化した辞書と深層強化学習を用いて、ロボットに「はめる」という動作を学習させる。具体的に学習させた内容は、

既に述べた通りロボットが白い円柱を掴み、赤いタブまで持っていくという課題である。本実験では ChainerRL 中の example として実装されていたものを我々の目的に合わせ、修正して使用した。

4.1 使用したロボット

本研究では, MuJoCo^{*1} [4] という商用物理シミュレータを使用した。またロボットアームは GitHub 上で公開されていた Pusher^{*2} を改変し 2 本の指を付け加えた 'Picker' を使用している。このロボットモデルは全部で 8 自由度を持ち、構成は図 3 のようになっている。

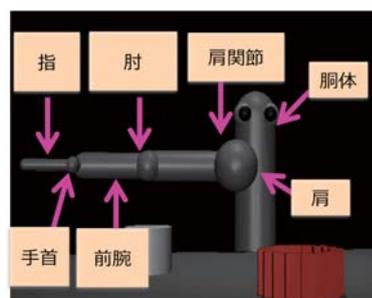


図 3: ロボットの構成図

4.2 実験設定

上記のロボットを OpenAIGym^{*3} というフリーの強化学習環境に組み込んで、ChainerRL^{*4} の上で深層強化学習を行なった。上述した A3C を用い、1 エピソードを 300 回の試行とし、全 80,000,000 エピソード学習させる実験を行なった。

円柱を掴んでほめるといった動作はいくつかの動作の構成要素から成り立つ。本研究では、それぞれの構成要素に対してその動作を実現するための行動価値関数を設定し、動作の系列に従ってそれらの行動価値関数を重み係数を持つヘビサイド関数を掛けて累加していくことにより、最終的な動作達成のための単独な行動価値関数を構築している。

具体的には

- 1 白い円柱に腕を近づける
- 2 白い円柱を掴む
- 3 白い円柱を赤いタブまで持っていく

というような 3 つの段階に動作を細分化し、その各々の条件をクリアするまでその後の行動の分が報酬に寄与しないように行動価値関数を設計した。

4.3 実験結果

以下に実験結果を示す。それぞれ取り上げたエピソード番号は報酬の値がそれまでで最も高かったものを取り上げている。

- 100,132 エピソード学習時 (図 4 参照)

100,132 エピソード学習時のモデルは以下ようになった。初期の状態から上腕を y 軸方向に回転させた状態で、少し腕を下げる動作をしていた。その後白い円柱に腕全体を持っていく動作を行っていた。掴むという動作をしないまま円柱を赤い

*1 <http://www.mujo.org/>

*2 <https://github.com/openai/gym/pull/557>

*3 <https://github.com/openai/gym>

*4 <https://github.com/chainer/chainerl>

タブに持って行ってしまい、「掴む」という動作の学習は上手くできていなかった。

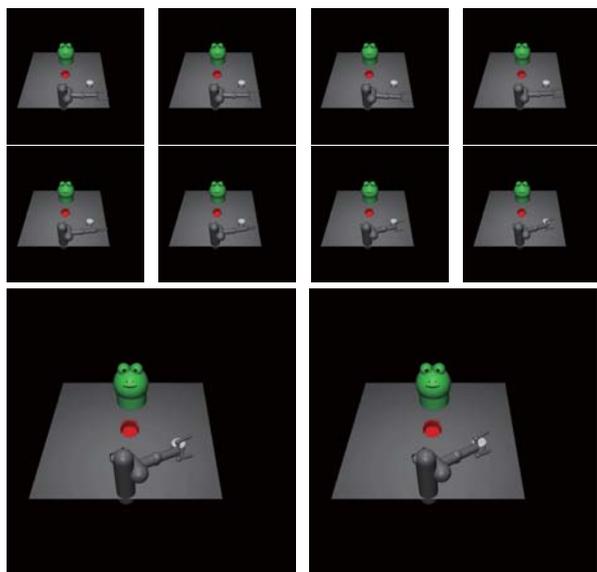


図 4: 100,132 エピソード学習時

● 200,327 エピソード学習時 (図 5 参照)

200,327 エピソード学習時のモデルは以下ようになった。初期の状態から腕を平行に移動させ、白い円柱の上まで持っていく様子が確認された。しかし、その後腕を下に降ろすことができなく、円柱を掴むことはできなかった。

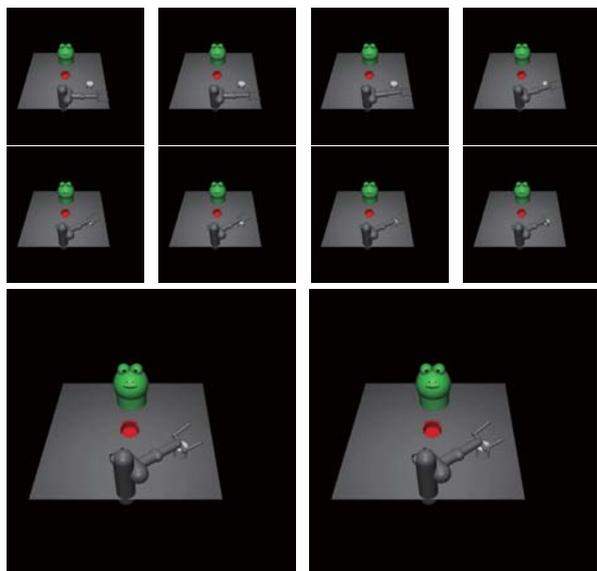


図 5: 200,327 エピソード学習時

● 300,570 エピソード学習時 (図 6 参照)

300,570 エピソード学習時のモデルは以下ようになった。初期の状態からすぐに腕を平行に移動させ、白い円柱の上まで

持っていき、その状態から腕を下ろすことを確認した。しかし、腕を下ろして掴んだ後に白い円柱を動かすような動作は見られなかった。

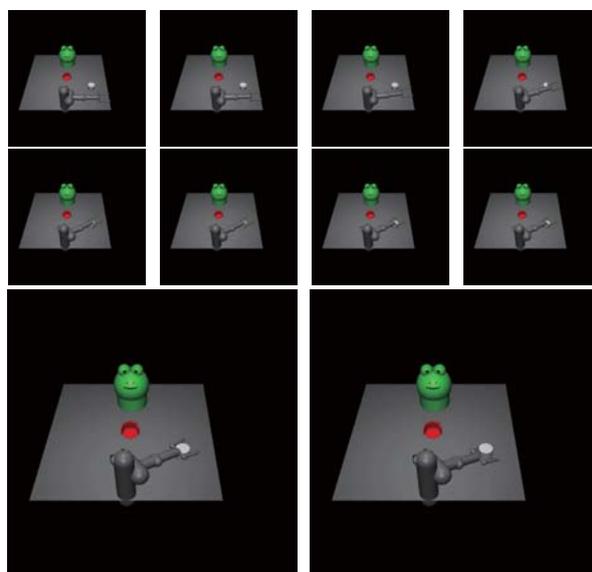


図 6: 300,570 エピソード学習時

● 1,100,031 エピソード学習時 (図 7 参照)

1,100,031 エピソード学習時のモデルは以下ようになった。初期の状態からすぐに腕を平行に移動させ、白い円柱の上まで持っていき、その状態から平行に下に腕を下ろし、掴んでいる様子が確認できた。掴んだ後に白い円柱に赤いタブを近づける動作も見られた。しかし、指が赤いタブに当たってしまい、しっかりとめはめることはできなかった。

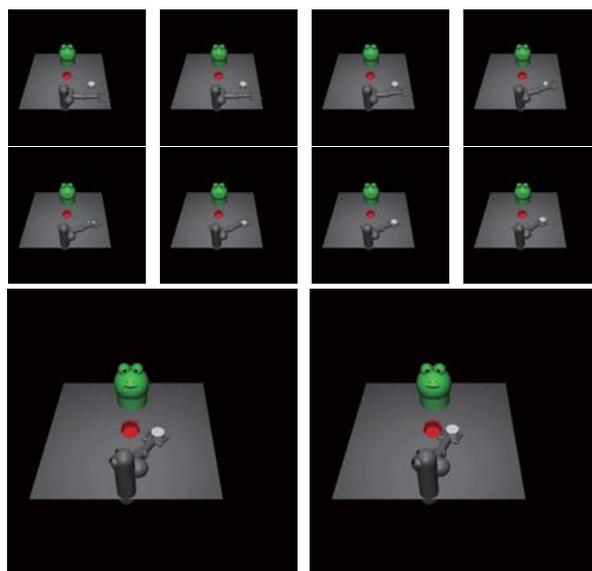


図 7: 1,100,031 エピソード学習時

● 4,600,083 エピソード学習時 (図 8 参照)

4,600,083 エピソード学習時のモデルは以下のようになった。初期の状態から腕を滑らかに平行に持っていき、腕を下に下ろして掴んだ後に白い円柱を赤いタブに近づける動作を確認できた。赤いタブに最も近づけていたが、一時的に円柱をはめた後にそこから取り出そうとするような動作を行っていた。

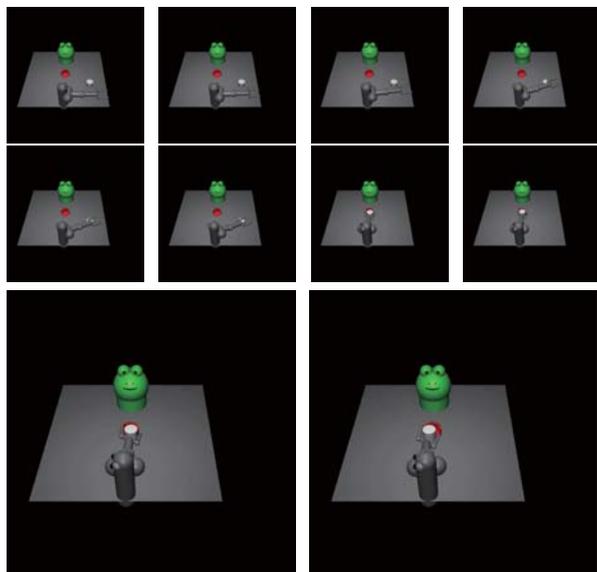


図 8: 4,600,083 エピソード学習時

4.4 考察

白い円柱を掴んで赤いタブにはめるまでの動作を一連の動作として学習させた。「はめる」という動作は腕を移動し、白い円柱を掴んで、赤いタブまで持っていくという動作を組み合わせることで構成されているが、この動作を一連の動作として学習させるために、行動価値関数を動作の構成要素に従って重み係数を掛けて累加していくという工夫を行なった。はめるという動作は人間にとっては簡単のように感じるが、ロボットにとっては複雑な動作であることがわかった。

5. まとめ

本研究では、動作を離散化した辞書と深層強化学習を用いてロボットに「はめる」という動作を学習させることを試みた。動作を離散化し辞書化することにより、学習時間の短縮と言語との対応付けの容易化を試みた。具体的には「はめる」という動作を学習させた。この動作は複雑であることから報酬を白い円柱に腕を近づける、白い円柱を掴む、白い円柱を赤いタブまで持っていく、という3段階に分けて考え、それらの動作に対する行動価値関数を重み付き係数を掛けて累加させることによって得られる単独の行動価値関数を用いて学習させた。今後の展望として、今回は白い円柱の初期位置を一定にして学習を行なっているが、今後は白い円柱の初期位置をランダムに与えて学習させたいと考えている。また、今回細分化して学習させた動作を別の動作の学習の際に使用するなど転移学習に応用できるかどうか試みたい。

参考文献

[1] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Tim Harley, Timothy P. Lillicrap, David Sil-

ver, Koray Kavukcuoglu, “Asynchronous methods for deep reinforcement learning”, PMLR 48:1928-1937, 2016.

- [2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Daan Wierstra, Alex Graves, Ioannis Antonoglou and Martin Riedmiller, “Playing Atari with Deep Reinforcement Learning”, In NIPS Deep Learning Workshop. 2013.
- [3] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver and Daan Wierstra, “Continuous control with deep reinforcement learning”, ICLR2016.
- [4] Emanuel Todorov, Tom Erez and Yuval Tassa, “MuJoCo: A physics engine for model-based control” IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012.