人工知能学会研究会資料
SIG-AGI-007-02

方策最適化による強化学習を用いた人型ロボットの動作学習の実験

Experiments on Motion Learning of Humanoid Robot with Reinforcement Learning by Policy Optimization

疋田 聡[1]

Satoshi Hikida[1]

[1]株式会社リコー
[1]Ricoh Company, LTD.

Abstract: Experiments on reinforcement learning were conducted on games on OpenAI Gym and robot simulators using "Proximal Policy Optimization Algorithms", which is considered to be suitable for motion learning of humanoid robots. As a result, it was confirmed that reinforcement learning is possible by the program of the algorithm published from OpenAI. Moreover, we confirmed that the operation on the robot simulator can be operated with real robot by the experimental experiment with real robot.

1. 背景

エージェントが自ら行動を決定し学習を進めていく強化学習は、汎用人工知能を実現するための有力な方法として盛んに研究されている。強化学習の代表的な手法としては、DQN[3][4]があるが、エージェントが取る行動がロボット制御のように連続値の場合は、A3C[2]や TRPO[5]のように方策最適化を行う手法の方が向いている。A3C は”Asynchronous Advantage Actor-Critic”の略称で、並列計算、Advantage と呼ばれる報酬の計算を 1step 後ではなく数ステップ後の報酬まで考慮した計算手法、方策関数を価値関数から分離した Actor-Critic モデルを用いることに特徴がある。また、TRPO は” Trust Region Policy Optimization” の略称で、方策関数が一度に大きく更新されすぎないように更新前と更新後の方策関数の KL ダイバージェンスの大きさで制限して、学習を安定化させる手法である。しかし、TRPO には実装が複雑になる、Actor-Critic モデルで方策関数と価値関数のパラメータを共有するのが難しいなどの欠点があった。

このような問題に対応する手法として、最近 OpenAI より PPO(Proximal Policy Optimization Algorithms)[6]というアルゴリズムが提案された。PPO は、更新前と更新後の方策関数の変化量が大きくなり過ぎないように、変化量が大きくなる場合には、変化量を一定の値にしてしまうというクリッピング操作を行うことで、学習を安定化させていることに特徴がある。方策関数が一度に大きく更新されすぎないようにするという考え方は TRPO と同じであるが、KL ダイバージェンスで制限する TRPO よりも実装が簡単になっている。

この PPO は、OpenAI よりその実装も公開されたので、OpenAI Gym[1]上のゲームとロボットシミュレータ上での学習実験により、その動作を確認してみることにした。また、ロボットシミュレータ上の動作が、実ロボットで動作可能かどうかについても実ロボットでの動作実験により確認してみることにした。

2. Proximal Policy Optimization Algorithms

文献[6]に報告されている、PPO アルゴリズムについて、より詳しく説明する。

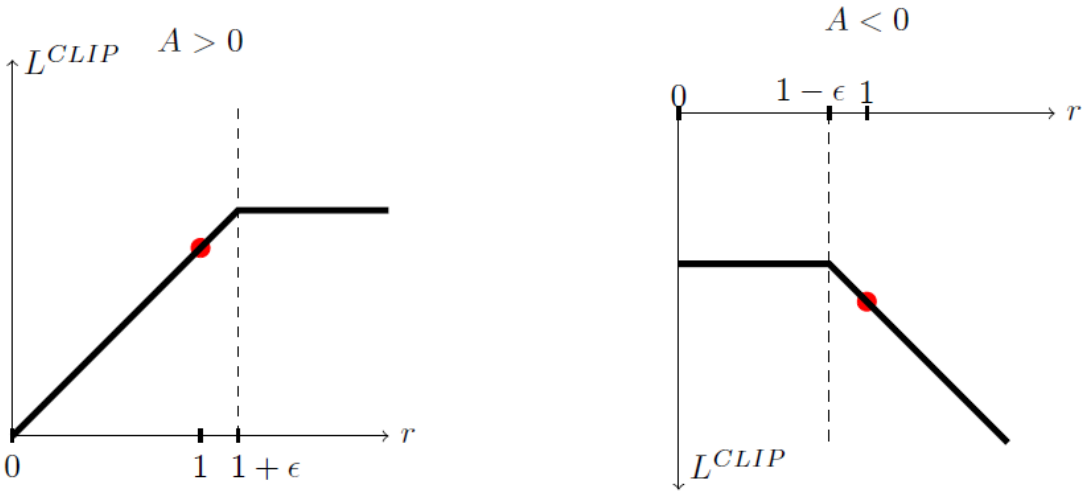

図 1：クリッピング操作の概念図(文献[6]の Figure 1 より引用)

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}$$

式 1：確率比の式（文献[6]の p.3 より引用）

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t\right] = \hat{\mathbb{E}}_t\left[r_t(\theta)\hat{A}_t\right]$$

式 2：の式（文献[6]の式(6)より引用）

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right]$$

式 3：の式（文献[6]の式(7)より引用）

この手法では、確率比を式 1 とすると、TRPO では式 2 を KL ダイバージェンスの大きさで制限しながら最大化しているが、PPO では式 3 のように変化量が大きくなる場合には、変化量を一定の値にしてしまうというクリッピング操作を行っている。文献[6]では、 ε は 0.2 が用いられている。図 1 にクリッピング操作の概念図を示す。図では、確率比が 1+ ε より大きい領域や 1- ε より小さい領域で L の値が大きくなり過ぎないように一定値でクリッピングされている様子が分かる。

このように、方策関数が一度に大きく更新されすぎないようにして学習を安定化させるという考え方は TRPO と同じであるが、KL ダイバージェンスで制限する TRPO よりも実装が簡単になっている。また、Actor-Critic モデルと組み合わせも可能となっている。

また、この手法は、OpenAI よりその実装も公開されている。

3. OpenAI Gym での学習実験

PPO アルゴリズムは、OpenAI よりその実装が公開されているので、OpenAI Gym 上のゲームでの学習実験により、その動作を確認してみた。

3.1. 実験方法

以下に実験方法の詳細を記載する。

PPO のアルゴリズムは、OpenAI より実装が公開されたものをそのまま用いた。また、ハイパーパラメータも公開されたものをそのまま用いた。

この実験は動作の確認を目的とするので、学習が比較的簡単なため良く用いられる OpenAI Gym 上の”CartPole”を用いて実験を行った。

3.2. 実験結果

以下に OpenAI Gym 上のゲームでの実験結果として報酬の推移のグラフを示す。

300 試行程度で学習できていることが確認できる。

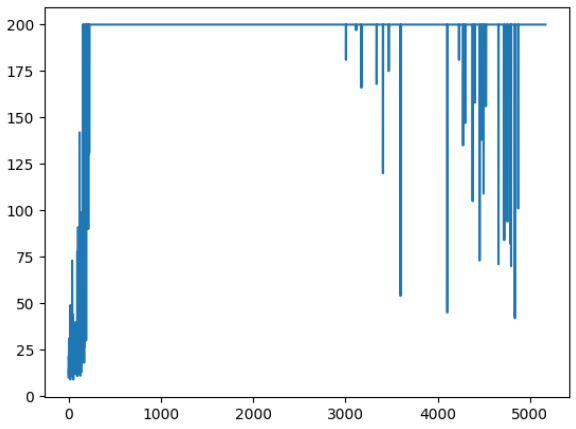

図 2：CartPole の強化学習の報酬の推移

4. ロボットシミュレータでの学習実験

OpenAI Gym 上のゲームでの動作は確認できたので、次により難しい課題としてロボットシミュレータ上での学習実験により、その動作を確認した。

4.1. 実験方法

ロボットシミュレータとしては、ROS(Robot Operating System)と Gazebo を用いた。

ロボットのモデルは、本来であれば実ロボットと同じものを用いる方がよいが、今回は時間の関係上実ロボット”DARWIN-MINI”と形状特徴が似ている”DARwIn-OP”のモデルで代用することにした。(GitHub HumaRobotics より取得可能である。)

なお、”DARwIn-OP”のモデルでの動作が”DARWIN-MINI”で実行可能かどうかの確認実験も行ったので、それについては 5 章に記載する。

ロボットシミュレーション環境と強化学習アルゴリズムの間のインタフェースは、OpenAI Gym の env 環境に合わせて作成した。また、学習させようとした動作は左方向への横歩きで、報酬などは以下のように設定した。

観測値は、各関節の角度と角速度とした。

行動は、各関節の角度の変化量とした。

報酬は、倒れていないことへの報酬、左方向への移動速度の報酬、前後方向への移動量のペナルティ、方向変化に対するペナルティ、行動の指示の 2 乗和のペナルティから計算した。

4.2. 実験結果

以下にロボットシミュレータでの実験結果として報酬の推移のグラフを示す。

図 3 の結果を見ると、学習が進むにつれて報酬が大きく下がっている部分が減ってきているので、ロボットが転倒しないことを学習していることが分かる。また、報酬の最大値の上がり方は緩やかなので、左方向への横歩きについてはまだ時間がかかりそうであることが読み取れる。

また、ロボットシミュレータ上での動きを画面表示で確認すると、グラフ上から読み取れる結果と同様になっていた。

この結果より、少なくともロボットが転倒しないことが学習できているので、PPO で学習が行われていることが確認できた。

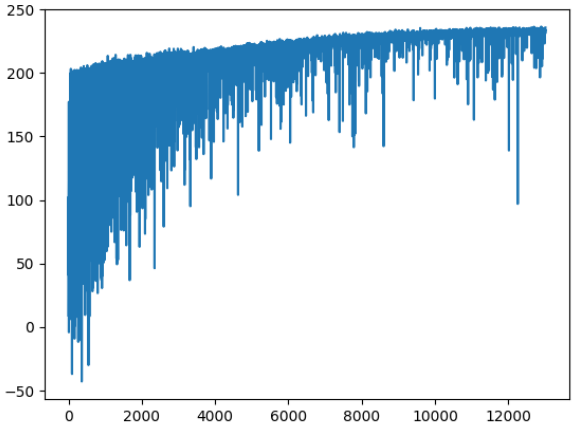

図 3：ロボットシミュレータ上での強化学習の報酬の推移

5. 実ロボットでの動作実験

ロボットシミュレータ上の動作が、実ロボットで動作可能かどうかについても実ロボットでの動作実験により確認してみた。

また、今回は時間の関係上実ロボットと形状特徴が似ている別のモデルで代用しているので、そのような運用でも実行可能かどうかの確認にもなっている。

5.1. 実験方法

実ロボットは”DARWIN-MINI”を用い、V-Sido CONNECT RC を接続してシリアル通信により各関節の制御情報を送れるようにした。

ロボットシミュレータは、ROS(Robot Operating System)と Gazebo を用いた。

ロボットのモデルは、実ロボットの”DARWIN-MINI”と形状特徴が似ている”DARwIn-OP”のモデルを用いた。

動作の実行は、まずロボットシミュレータ上でロボットの動作を行ない、その動作時の各関節の角度を 50ms 毎に記録し、そのデータを実ロボットへ送って動作が再現させることにより行った。

なお、強化学習している動作は学習に時間がかかっているので、”DARwIn-OP”のモデルの環境に実装されていた横歩きの動作で確認を行うことにした。

5.2. 実験結果と考察

”DARwIn-OP”のモデルの環境に実装されていた横歩きの動作を記録し、実ロボットの”DARWIN-MINI”に送って動作させたところ、うまく動作させることができた。これは、実ロボットとシミュレータモデルでロボットの大きさは異なるが、

全体の形状の比率が類似しているため、動作時の関節角のデータも類似したものとなり、関節角のデータを送れば同じ動作が可能になっていると考えられる。

6. まとめと今後

人型ロボットの動作学習に適していると考えられる、PPO アルゴリズムについて、OpenAI Gym 上のゲームとロボットシミュレータ上での学習実験により、その動作を確認できた。

また、ロボットシミュレータ上の動作が、実ロボットで動作可能なことを実ロボットでの動作実験により確認できたので、今回は時間の関係上実施できなかったが、今後はロボットシミュレータ上での学習結果を実ロボットで動作させていく予定である。

将来的には、強化学習手法の進歩、ロボットハードの低価格化、強化学習が可能なシミュレータ環境の利用などにより、人型ロボットの動作学習が容易になり、人型ロボットの活用可能性も増えていくと思われる。

参考文献

[1] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W.: OpenAI gym, arXiv preprint arXiv:1606.01540, (2016)

[2] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., ... & Kavukcuoglu, K.: Asynchronous methods for deep reinforcement learning, In International Conference on Machine Learning (pp. 1928-1937), (2016)

[3] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M.: Playing atari with deep reinforcement learning, arXiv preprint arXiv:1312.5602, (2013)

[4] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S.: Human-level control through deep reinforcement learning, Nature 518(7540) 529-533, (2015)

[5] Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P.: (2015). Trust region policy optimization. In Proceedings of the 32nd International Conference on Machine Learning (ICML-15) (pp. 1889-1897).

[6] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O.: Proximal policy optimization algorithms, arXiv preprint arXiv:1707.06347, (2017)