特別研究報告書

模倣学習による
修士論文生成手法の検討

指導教員　　森本淳　教授
八木聡明　助教

京都大学情報学研究科
システム科学専攻
令和 4 年 4 月入学

古巻　鉄平

令和 6 年 2 月 5 日提出

摘要

目次

1 **序論** 1

2 **関連研究** 1

2.1 スキルの事前学習？ 1

2.2 多指ハンドロボット 2

2.3 接触の多いタスクの制御 2

3 **手法** 2

3.1 強化学習 (整理し直す) 2

3.2 Soft Actor-Critic 法（分割する） 3

3.3 変分オートエンコーダー 5

3.4 スキルとその事前分布の学習 5

4 **実験** 5

4.1 実験のセットアップ 5

4.2 予備実験：放物線追従タスク 6

4.3 実験 1：同一形状のバルブの学習 6

4.4 実験 2：異なる形状のバルブへの転移 7

4.5 実験 3：同一形状のバルブにおける実機への転移 7

5 **議論** 7

6 **結論** 7

参考文献 7

1 序論

人間の手を模倣した多指ハンドロボットの実用化は社会的に大きな意義がある．人間が日常生活を送るにあたって手で物体を操るという動作は必要不可欠であるが，多指ハンドロボットが実現することでこれらの動作を代替することができる．このような多指ハンドロボットには，多種多様な形状を持つ物体を操る能力が要求される．これまで様々な多指ハンドロボットの開発が進められてきたものの，多指ハンドロボットを人間のように器用に動かす制御手法については依然として開発が進んでおらず実用化とは程遠い状況にある．

多指ハンドロボットの制御手法の開発では，操作する物体との間に接触が多くモデル化が困難である点が特に問題となる．このような課題を解決するため，多指ハンドロボットの制御手法としてモデルフリー強化学習がよく用いられる．モデルフリー強化学習では力学モデルの存在を陽に仮定せず，環境との相互作用を繰り返すことによって制御方策を獲得する [1]．一方で，モデルフリー強化学習は複雑なタスクの学習を可能にしたものの，あるタスクを学習して得られた方策を環境のわずかに異なる別のタスクに適用するのは困難である [2]．このため，多指ハンドロボットの目的である様々な形状の物体の操作を達成することが出来ない．

このようなモデルフリー強化学習の課題を解決するために（12 月 8 日ミーティングでの話を考慮して書く．「階層強化学習」ではない．「オプション」「downstream task」あたりの話をまとめて書く．）

本研究ではこのような〜〜〜を用いて複数の形状のバルブを短時間で学習させる．

(実験結果)

2 関連研究

2.1 スキルの事前学習？

事前に収集したデータから複数のタスクの学習を効率化するための手段として，スキルの事前学習が研究されている．スキルの事前学習では，情報が付加されていないデータを収集し，そのデータからスキルを抽出する．その後，抽出したスキルを行動として利用することで学習する．Singh らは，正規化フロー法 [3] を用いて標準正規分布に従うノイズベクトルを行動ベクトルへ射影することで，学習速度を向上させる行動の事前分布を獲得した [4]．また，Pertsch らは，事前に収集したデータから行動の系列をサンプリングし，変分オートエンコーダ (VAE) によってサンプリングした系列の再構成学習を行うことで，スキルを潜在空間上に埋め込み高速な学習を実現する手法を開発した [5]．本研究では事前に収集したデータからスキルを抽出することでハンドロボットの回転タスクについて，未知のタスクに対する学習速度を向上させる手法を検討する．

2.2 多指ハンドロボット

・多指ハンドロボットは古くから開発されてきた Adroit[6] など？
・近年は Shadow など本格的なロボットも
・一方で，低コストなトイロボットである Robel[7]，LEAP hand[8] などが強化学習の研究に向く．

2.3 接触の多いタスクの制御

・強化学習を用いている研究をいくつか
・目的は，現状どこまで出来ているかを明らかにすること．
・OpenAI のルービックキューブの研究など．

3 手法

3.1 強化学習 (整理し直す)

本研究ではロボットの制御問題をマルコフ決定過程 $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p, r)$ によってモデリングする．$\mathcal{S}$，$\mathcal{A}$ はそれぞれ状態空間，行動空間を意味しており，現在の状態 $s_t \in \mathcal{S}$ で行動 $a_t \in \mathcal{A}$ をとった時に状態 $s_{t+1} \in \mathcal{S}$ へと遷移する状態遷移確率を $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$，この時エージェントが得る報酬を $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ と表す．また，方策 $\pi(a_t|s_t)$ に従って行動した際に得られる状態と行動の系列において，s_t が表れる確率を $\rho_\pi(s_t)$，s_t と a_t が同時に表れる確率を $\rho_\pi(s_t, a_t)$，デモンストレーションデータの中に s_t と a_t が同時に現れる確率を $\rho_D(s_t, a_t)$ と表記する．
通常の強化学習では，累積報酬 $\sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t)]$ を最大化することを目指す．ただし，実際はステップ数が無限大の時に発散しないために割引率 γ を r の前にかけることが多い．累積報酬の最大化に向け，まず各状態における状態価値関数と行動を含めた行動価値関数を以下のように定義する．

$$V_\pi(s) = \mathbb{E}_{\rho_\pi}\left[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k}) \middle| s_t = s\right] \tag{1}$$

$$Q_\pi(s, a) = \mathbb{E}_{\rho_\pi}\left[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k}) \middle| s_t = s, a_t = a\right] \tag{2}$$

この時，式 (1) を

$$\begin{aligned} V_\pi(s) &= \mathbb{E}_{\rho_\pi}\left[r(s_t, a_t) + \sum_{k=1}^{\infty} \gamma^k r(s_{t+k}, a_{t+k}) \middle| s_t = s\right] \\ &= \mathbb{E}_{\rho_\pi}\left[r(s_t, a_t) + \gamma V_\pi(s_{t+1}) | s_t = s\right] \end{aligned} \tag{3}$$

のように変形することでベルマン方程式が得られる．式 (3) における期待値をとる操作を最大値をとる操作に置き換えたものをベルマン最適方程式と呼び，期待値をとるものはベルマン期待方程式と呼ぶ．このベルマン方程式の右辺を $V_\pi(\cdot)$ の写像と見做すことで次のようなベルマン作用素 $\mathcal{T}^\pi$ を定義する．

$$\mathcal{T}^\pi \circ V(s) = \mathbb{E}_{\rho_\pi}\left[r(s_t, a_t) + \gamma V(s_{t+1}) | s_t = s\right] \tag{4}$$

式 (2) の状態価値関数についても同様にベルマン方程式やベルマン作用素を定義できる．特に，ベルマン期待方程式についての作用素をベルマン期待作用素，ベルマン最適方程式についての作用素をベルマン最適作用素と呼ぶ．ベルマン作用素は縮小写像であるため $V_\pi(s)$ はランダムな初期値から何度もベルマン作用素を作用させることにより真の関数に収束する．
このような価値関数を求めるための方法は，動的計画法，モンテカルロ法，TD 法の 3 種類に大別される．動的計画法は環境の情報が既知であると仮定した上でベルマン方程式を解く手法で，ベルマン期待作用素による方策の評価と改善を繰り返す方策反復とベルマン最適作用素を用いて最適な方策を求める価値反復に大別できる．モンテカルロ法はエージェントが探索して得られた経験をもとに学習を行う．エージェントが 1 エピソード探索するごとに方策や価値関数の更新を行う．TD 法は，動的計画法においてある状態の価値関数を推定する際に一つ前の状態の推定値を利用するという特性と，モンテカルロ法におけるエージェントの経験を利用するという特性を組み合わせたもので，1 ステップごとに更新を行う．TD 法を使ったアルゴリズムとして方策オン型の SARSA や方策オフ型の Q-leraning が存在する．
通常の強化学習は状態空間，行動空間が離散の場合，価値関数を表形式で保存することにより実行できるが，状態空間と行動空間が大きい場合や連続な場合，価値関数を表形式で表すと膨大なメモリを必要とする．ロボット制御では連続な物理系を考えるため価値関数は表形式ではなく近似器のパラメータとして保存し，最適化したい目的関数に応じて更新する．

3.2 Soft Actor-Critic 法（分割する）

Soft Actor-Critic(SAC)[9, 10] は連続な行動空間上で学習できる方策オフ型強化学習アルゴリズムである．このような Actor-critic アルゴリズムは方策を保存する Actor と価値関数を保存する Critic という二つのネットワークを持っておりこれらを更新することによって学習を行う．
状態価値関数は累積報酬に探索を促進するためのエントロピー項を加えた式 (5) で表される．この時，実際は報酬に割引率がかけられており，また，エントロピーの項に存在する α は温度パラメータである．温度パラメータは学習が進むにつれて小さくしてゆくことで最終的な方策を決定的な方策に近づける．

$$V(s_t) = \mathbb{E}_{a_t \sim \pi}\left[r(s_t, a_t) - \alpha \log \pi(a_t | s_t)\right] \tag{5}$$

また，行動価値関数は先ほどの状態価値関数を用いて，

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{a_t \sim p}[V(s_{t+1})] \tag{6}$$

のように表すことができる．これを式 (5) に代入することで

$$V(s) = \mathbb{E}_{a_t \sim \pi} \left[Q(s,a) - \alpha \log \pi(a|s) \right] \tag{7}$$

という式が得られる．一方，SAC において方策はガウス方策であるものとし，平均と分散によってパラメータ化する．このような方策を価値関数を最大にするよう以下のように更新を行う．なお，Π はガウス方策全体からなる集合を意味する．

$$\pi_{\text{new}} = \underset{\pi' \in \Pi}{\operatorname{argmin}} D_{KL} \left(\pi'(\cdot|s_t) \| \frac{\exp(Q^{\pi_{old}}(s_t, \cdot))}{Z^{\pi_{old}}(s_t)} \right) \tag{8}$$

次にこれらの価値関数や方策を関数近似器で表現する．soft-Q 関数は $Q_\theta(s_t, a_t)$，方策は π_ϕ のように表記する．今回の実験では関数近似器としてニューラルネットワークを用いるため，θ, ϕ はネットワークのパラメータを表す．

soft-Q 関数の損失関数を

$$J_Q(\theta) = \mathbb{E}_{(s_t, a_t) \sim D} \left[\frac{1}{2} \left(Q_\theta(s_t, a_t) - \left(r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p}[V_{\bar{\theta}}(s_{t+1})] \right) \right)^2 \right] \tag{9}$$

のように表す．ただし，$V_{\bar{\theta}}(s_{t+1})$ は soft-Q 関数から式 (7) を使って求める．また，$\bar{\theta}$ は target-Q 関数のパラメータで，soft-Q 関数のパラメータの指数加重移動平均として計算される．式 (9) の勾配は

$$\hat{\nabla} J_Q(\theta) = \nabla_\theta Q_\theta \Big(Q_\theta - \Big(r(s_t, a_t) + \gamma \big(Q_\theta - \alpha \log(\pi_\phi(a_{t+1}|s_{t+1})) \big) \Big) \Big) \tag{10}$$

のようになる．soft-Q 関数を実装する際は，SAC では方策改善における正のバイアスを軽減するため 2 つの Q 関数を同時に学習させ，ある状態行動対を評価する際にはこれらのうち値の小さい方を利用する [11]．そのため，target-Q 関数も合わせて合計で 4 つのネットワークで学習することとなる．方策の損失関数は式 (8) の KL ダイバージェンスを最小にするため以下のように計算する．

$$J_\pi(\phi) = \mathbb{E}_{s_t \sim D, a_t \sim \pi_\phi(\cdot|s_t)} \left[\log \pi_\phi(a_t|s_t) - Q_\theta(s_t, a_t) \right] \tag{11}$$

この損失関数は行動 a_t についての期待値を取っているが，a_t は $\pi_\phi(\cdot|s_t)$ からサンプリングしたものなので先ほどのように勾配を求めることはできない．$\pi_\phi(\cdot|s_t)$ はガウス分布であることから，ニューラルネットワークでガウス分布の平均と標準偏差を求めそれを基に a_t を構成する reparameterized trick によって勾配を求める．この時，a_t は標準正規分布からサンプリングした ε_t を用いて

$$a_t = f_\phi(\varepsilon_t; s_t) \tag{12}$$

と表すと, 式 (11) は

$$J_\pi(\phi) = \mathbb{E}_{s_t \sim D, \varepsilon_t \sim \mathcal{N}(0,1)} \left[\log \pi_\phi(f_\phi(\varepsilon_t; s_t)|s_t) - Q_\theta(s_t, f_\phi(\varepsilon_t; s_t)) \right] \tag{13}$$

のように変形でき, この勾配は

$$\hat{\nabla}_\phi J_\pi(\phi) = \nabla_\phi \alpha \log \pi_\phi(a_t|s_t) + \nabla_\phi f_\phi(\varepsilon_t; s_t) [\nabla_{a_t} \alpha \log \pi_\phi(a_t|s_t) - \nabla_{a_t} Q_\theta(s_t, a_t)] \tag{14}$$

として計算できる．また，エントロピー項にかかっている温度パラメータは以下のような目的関数の勾配を取ることによって更新を行う [10]．この時，$\bar{\mathcal{H}}$ は目標エントロピーを表している．

$$J(\alpha) = \mathbb{E}_{a_t \sim \pi_t}[-\alpha \log \pi_t(a_t|s_t) - \alpha\bar{\mathcal{H}}]. \tag{15}$$

3.3 変分オートエンコーダー

3.4 スキルとその事前分布の学習

Pertsch らが提案した SPiRL は事前に収集したデータからスキルを抽出し，それらの事前分布を学習することで学習の高速化を実現したアルゴリズムである [5]．このアルゴリズムの概要を図**??**に示す．この手法ではスキル $\boldsymbol{a_i}$ を H 個の行動の系列 $a_t^i, ..., a_{t+H-1}^i$ として定義し，これらのスキルを低次元の潜在空間 $\mathcal{Z}$ に埋め込む．スキルの埋め込みには β-VAE を用い，以下の式の右辺であるエビデンス下界（ELBO）を最大化することで学習する．

$$\begin{aligned} \log p(\boldsymbol{a_i}) \geq \mathbb{E}_q[&\log p(\boldsymbol{a_i}|z) \\ &- \beta(\log(z|\boldsymbol{a_i}) - \log p(z)] \end{aligned} \tag{16}$$

ここで，z の事前分布 $p(z)$ は標準正規分布に従っており，潜在空間の学習後は標準正規分布からサンプリングした z をデコーダーに入力することで行動の系列を得ることができる．
スキルの抽出と同時に，エージェントの学習を促進するためスキルの事前分布 $p_{\boldsymbol{a}}(z|s_t)$ を学習し，各状態 s_t において優先度の高いスキルの情報を抽出する．スキルの事前分布は Kullback-Leibler ダイバージェンス

$$\mathbb{E}_{(s,\boldsymbol{a}_i)\sim D} D_{KL}\left(q(z|\boldsymbol{a}_i), p_{\boldsymbol{a}}(z|s_t)\right) \tag{17}$$

が最小になるよう学習する．
スキルの抽出と事前分布の学習後，抽出したスキルを用いてタスクを学習する．この手法において方策は行動 a_t の代わりにスキルの潜在変数 z を出力する．出力された潜在変数 z を学習したデコーダーに入力し，そこから出力された行動の系列を環境に入力する．このような方策 $\pi_\theta(z|s_t)$ のパラメータのを通常の強化学習の枠組みを用いて更新する．

4 実験

4.1 実験のセットアップ

4.1.1 3 指ハンドロボット：D'Claw

本研究ではオープンソースロボットプラットフォーム Robel のハンドロボット D'Claw で実験する [7]．D'Claw の構造は図 1 に示しており，3 つの関節を備えた指を 3 本，合計で 9 つの関節が存在する．各指の関節の可動域は根元から $27.5°, 60°, 90°$ である．行動空間は 9 次元で，各関節の目標角度を可動域の大きさで $[-1, 1]$ の範囲にスケールしたものを入力とする．状態空間は以

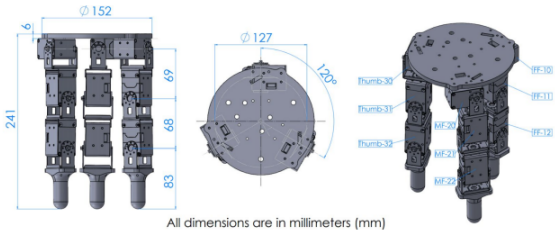

図 1: D'Claw の概要．FF-10∼Thumb-32 までの角度を $\theta_{\mathrm{j}}^{1} \sim \theta_{\mathrm{j}}^{9}$ とし，$\boldsymbol{\theta}_{\mathrm{j}} = (\theta_{\mathrm{j}}^{1}, \cdots, \theta_{\mathrm{j}}^{9})$ とおく．また，バルブの角度を θ^{v} のように表記する．[7] より引用．

下のように関節の角度 $\boldsymbol{\theta}^{\mathrm{j}}$，関節の角速度 $\dot{\boldsymbol{\theta}}^{\mathrm{j}}$，バルブの角度の余弦 $\cos(\theta^{\mathrm{v}})$，バルブの角度の正弦 $\sin(\theta^{\mathrm{v}})$，バルブの角度と目標角度の誤差 $\theta^{\mathrm{v}} - \hat{\theta}^{\mathrm{v}}$ の合計 21 次元からなる．

$$s_t = \left(\boldsymbol{\theta}^{\mathrm{j}}, \dot{\boldsymbol{\theta}}^{\mathrm{j}}, \cos(\theta^{\mathrm{v}}), \sin(\theta^{\mathrm{v}}), \theta^{\mathrm{v}} - \hat{\theta}^{\mathrm{v}}\right) \tag{18}$$

4.1.2 シミュレーション環境

このロボットをシミュレーションで学習するにあたって，オープンソースの物理エンジンである MuJoCo を利用する [12].

4.1.3 実機の設定

アクチュエータとして Dynamixel XM430-W210 を利用する．

4.2 予備実験：放物線追従タスク

アルゴリズムを検証するにあたって，接触の存在しない環境における学習性能を検証する．

4.3 実験 1：同一形状のバルブの学習

・データの存在するものと同一形状のバルブについて学習．

4.4 実験 2：異なる形状のバルブへの転移

4.5 実験 3：同一形状のバルブにおける実機への転移

5 議論

6 結論

謝辞

に深く感謝する.

参考文献

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning : an introduction*. MIT Press, 2nd ed edition.

[2] Jiang Hua, Liangcai Zeng, Gongfa Li, and Zhaojie Ju. Learning for a robot: Deep reinforcement learning, imitation learning, transfer learning. *Sensors*, Vol. 21, No. 4, p. 1278, 2021.

[3] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[4] Avi Singh, Huihan Liu, Gaoyue Zhou, Albert Yu, Nicholas Rhinehart, and Sergey Levine. Parrot: Data-driven behavioral priors for reinforcement learning. *arXiv preprint arXiv:2011.10024*, 2020.

[5] Karl Pertsch, Youngwoon Lee, and Joseph Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on robot learning*, pp. 188–204. PMLR, 2021.

[6] Vikash Kumar, Yuval Tassa, Tom Erez, and Emanuel Todorov. Real-time behaviour synthesis for dynamic hand-manipulation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6808–6815. IEEE, 2014.

[7] Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. Robel: Robotics benchmarks for learning with low-cost robots. In *Conference on robot learning*, pp. 1300–1313. PMLR, 2020.

[8] Kenneth Shaw, Ananye Agarwal, and Deepak Pathak. Leap hand: Low-cost, efficient, and anthropomorphic hand for robot learning. *arXiv preprint arXiv:2309.06440*, 2023.

[9] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1861–1870.

PMLR. ISSN: 2640-3498.

[10] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications.

[11] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1587–1596. PMLR. ISSN: 2640-3498.

[12] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. ISSN: 2153-0866.

※線に沿って切り取って下さい。

特別研究報告書

模倣学習による
修士論文生成手法の検討

指導教員　　森本淳　教授
　　八木聡明　助教

京都大学情報学研究科
システム科学専攻
令和 4 年 4 月入学

古巻　鉄平

令和 6 年 2 月 5 日提出

模倣学習による修士論文生成手法の検討

古巻　鉄平

令和５年度

模倣学習による
修士論文生成手法の検討

古巻　鉄平

摘要