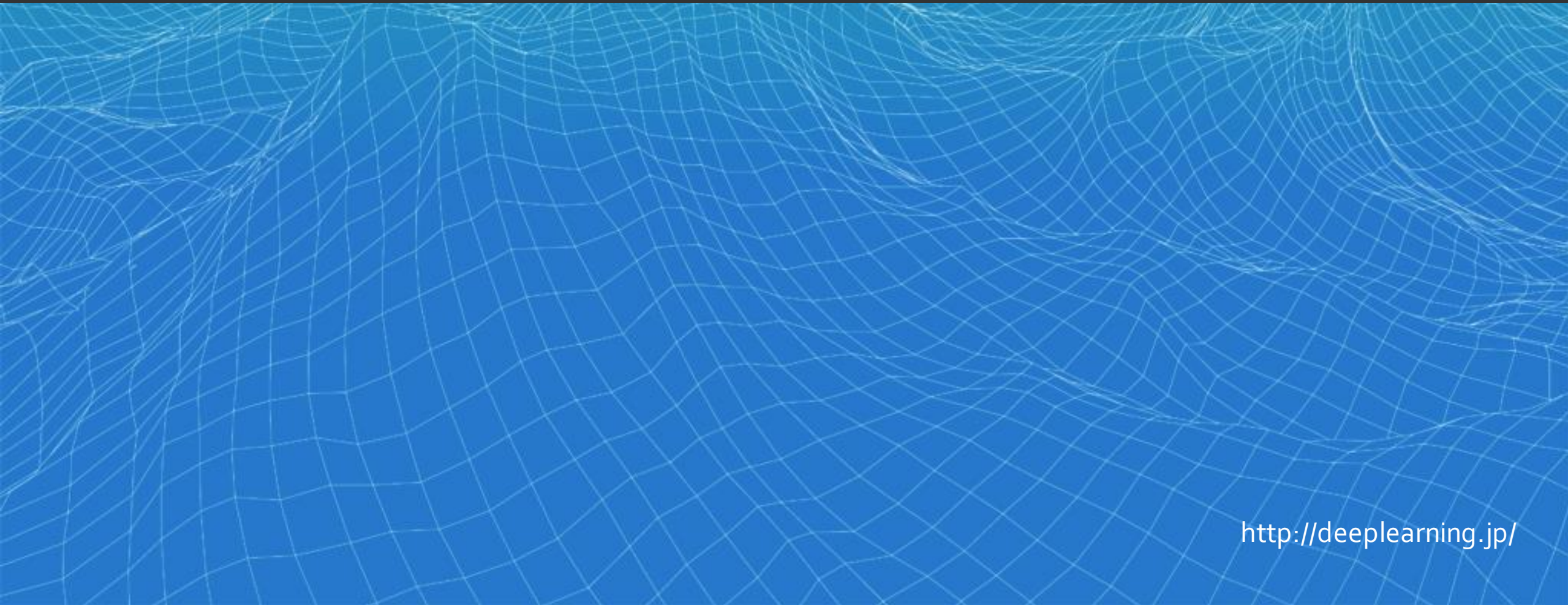
DEEP LEARNING JP
[DL Papers]

Stabilizing Reinforcement Learning in Differentiable Multiphysics Simulation

Ku Onoda, Matsuo-Iwasawa Lab, M1



http://deeplearning.jp/

書誌情報

- Stabilizing Reinforcement Learning in Differentiable Multiphysics Simulation
  - ICLR 2025 (Spotlight)

- 著者
  - Eliot Xing, Vernon Luk, Jean Oh
- リンク
  - https://rewarped.github.io/

概要

- 課題
  - 変形体（ロープ・布・流体・粘土など）は剛体より計算負荷が高く、従来のRLではサンプル効率が致命的に低い

- 本研究の貢献
  1. Rewarped：GPU並列・微分可能なマルチフィジックス環境のシミュレータ
  2. SAPO：最大エントロピー×解析勾配の FO-MBRL
  3. 様々なタスクで従来の手法（PPO/SAC/APG/SHAC）を上回る

背景①　RL × 物理シミュレーション

- RL　累積期待報酬を最大化する方策π_θのパラメータθを最適化
  - 目的関数 $$J(\pi) = \mathbb{E}_{\substack{s_0 \sim \rho_0 \\ \tau \sim \rho_\pi}} [R_{0:T}]$$
  - 歩行・把持・マニピュレーションなどのロボット制御への応用がある

- シミュレータの役割
  - 実機でのデータ収集は高コスト　→　シミュレーションで大量のデータ生成
  - GPU並列環境により剛体タスクは数時間で学習が可能

- 現状の課題
  - 変形体（deformable objects）では計算が2-3桁遅い
  - モデルフリーRLでは10^8 step以上要求され現実的でない

背景② 微分可能シミュレータ

- 普通のシミュレータ
  - 遷移関数や報酬関数の内部がブラックボックス
  - 勾配の0次推定量

$$\hat{\nabla}_\theta^{[0]} J(\pi) = R_{0:T} \sum_{t=0}^{T-1} \nabla_\theta \log \pi(\boldsymbol{a}_t|\boldsymbol{s}_t)$$

- 微分可能シミュレータ
  - シミュレータ内部の遷移関数・報酬関数内の計算を微分可能な演算のみに限定
  - 方策のパラメータに対して累積報酬の勾配を解析的に利用可能

    → より効率的で安定した学習を可能にする

$$\hat{\nabla}_\theta^{[1]} J(\pi) = \nabla_\theta R_{0:T}$$

背景③　既存シミュレータ比較

Simulator	∇?	Materials?				
		Rigid	Articulated	Elastic	Plasticine	Fluid
Isaac Gym	✗	✓	✓	✓	✗	✗
Isaac Lab / Orbit	✗	✓	✓	✓*	✗	✓*
ManiSkill	✗	✓	✓	✗	✓*	✓*
TinyDiffSim	✓	✓	✓	✗	✗	✗
Brax	✓	✓	✓	✗	✗	✗
MJX	✓*	✓	✓	✗	✗	✗
DaXBench	✓	✓	✗	✗	✓*	✓
DFlex	✓	✓	✓	✓*	✗	✗
Rewarped (ours)	✓	✓	✓	✓	✓	✓

- 「変形体対応」「微分可能」「並列化可能」全てを満たすものはない

論文の貢献

1. シミュレータ

   Rewarped：WarpベースでGPU上に全環境をバッチ展開

   逆モード自動微分+CUDA Graphによる高速勾配計算

2. アルゴリズム

   SAPO：最大エントロピー FO-MBRL（First-Order Model Base RL）

   →探索と安定性を両立

3. 実験的検証

   剛体 2 タスク + 変形体 4 タスク

   全タスクで既存手法を上回る（最大で7倍）

Rewarped　コンセプト

- Rewarped: Warp（NVIDIAが公開するGPU向けDSL）をもとに実装されたシミュレーションプラットフォーム

- Warpとの差分
  - 既存のWarpの実装は”単一環境”前提
  - Rewarpedは N環境 × Tステップ をバッチ並列化

- 対応マテリアル
  - 剛体・関節・弾性体
  - 粘土
  - 流体

Rewarped　実装の工夫①

- Material Point Method（MPM）
  - 変形体を粒子+格子で解く方法
  - 既存のMLS-MPM手法（Ma et al., 2023）は単一環境向け
  - **粒子データをバッチ化**することでN環境を同時に計算

- 剛体との相互作用
  - 一方向結合：関節剛体の衝突形状を**kinematics**（外力に影響されない）としてMPM粒子に力を与える
  - 逆方向の力は無視することで安定性を確保し、実用上の多くのマニピュレーションタスクをカバー

Rewarped　実装の工夫②

- CUDA Graph
  - forward/backward をテープに記録
  - 後から”adjointカーネル”を自動生成し、逆伝播を実行
  - ソースコード変換（Griewank & Walther, 2008）で解析的勾配を取得
    - 有限差分法よりも高速かつ高精度
- Gradient Check-pointing
  - 順伝播では中間値を保存せず、逆伝播時に該当ステップを再計算
- PyTorch Autograd ブリッジ
  - Warp 側テンソルを PyTorch Tensor と共有
  - RLアルゴリズムは通常のPyTorchコードで記述可
  - シミュレータ側はGPUネイティブで高速

SAPO　コアの考え

- FO-MBRL
  - シミュレータが微分可能なら勾配をREINFORCEではなく解析勾配として直接取得可能

- 課題：解析勾配は、目的関数の非連続性やカオス的挙動により学習が不安定となる
  - 特に、接触が伴うタスク、長いホライズン

- 方向性：報酬にエントロピー項を加える
  - LandscapeをSmoothing
  - 探索も促進 → 局所最適を回避

既存手法①　FO-MBRL

- Analytic Policy Gradient（APG, Freeman et al. (2021)）
  - 直接割引報酬を最大化する
  - horizonがエピソード全ての場合はBPTTと呼ばれるもの

$$J(\pi) = \sum_{l=t}^{t+H-1} \mathbb{E}_{(\boldsymbol{s}_l, \boldsymbol{a}_l) \sim \rho_\pi}[\gamma^{l-t} r_l]$$

- Short Horizon Actor Critic （SHAC, Xu et al. (2021)）
  - 方策π_θと(ホライズンの終端の)価値関数V_ψを学習

$$J(\pi) = \sum_{l=t}^{t+H-1} \mathbb{E}_{(\boldsymbol{s}_l, \boldsymbol{a}_l) \sim \rho_\pi}[\gamma^{l-t} r_l + \gamma^t V(\boldsymbol{s}_{t+H})] \qquad \mathcal{L}(V) = \sum_{l=t}^{t+H-1} \mathbb{E}_{\boldsymbol{s}_l \sim \rho_\pi}[||V(\boldsymbol{s}) - \tilde{V}(\boldsymbol{s})||^2]$$

  - $\tilde{V}$はTD(λ)を使って計算

$$\tilde{V}(\boldsymbol{s}_t) = G^\lambda_{t:t+H} = (1-\lambda)\left(\sum_{l=1}^{H-1-t} \lambda^{l-1} G_{t:t+l}\right) + \lambda^{H-t-1} G_{t:t+H}$$

  - 1次勾配

$$\hat{\nabla}^{[1]}_\theta J(\pi) = \nabla_\theta (R_{0:H} + \gamma^H V(\boldsymbol{s}_H))$$

既存手法②　エントロピー最大化RL

- SAC（Soft Actor-Critic）
  - モデルフリーの連続行動空間に対応するRLアルゴリズム

$$J(\pi) = \sum_{t=0}^{\infty} \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t) \sim \rho_\pi} [r_t + \alpha \mathcal{H}_\pi[\boldsymbol{a}_t | \boldsymbol{s}_t]] \quad \text{where } \mathcal{H}_\pi[\boldsymbol{a}_t | \boldsymbol{s}_t] = -\int_{\mathcal{A}} \pi(\boldsymbol{a}_t | \boldsymbol{s}_t) \log \pi(\boldsymbol{a}_t | \boldsymbol{s}_t) d\boldsymbol{a}_t$$

割引率を導入

$$J_{\text{maxent}}(\pi) = \sum_{t=0}^{\infty} \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t) \sim \rho_\pi} \left[\sum_{l=t}^{\infty} \gamma^{l-t} \mathbb{E}_{(\boldsymbol{s}_l, \boldsymbol{a}_l) \sim \rho_\pi} [r_t + \alpha \mathcal{H}_\pi[\boldsymbol{a}_l | \boldsymbol{s}_l]] \right]$$

この時のQ値と価値関数V

$$Q^\pi_{\text{soft}}(\boldsymbol{s}_t, \boldsymbol{a}_t) = r_t + \mathbb{E}_{(\boldsymbol{s}_{t+1}, \ldots) \sim \rho_\pi} \left[\sum_{l=t+1}^{\infty} \gamma^l (r_l + \alpha \mathcal{H}_\pi[\boldsymbol{a}_l | \boldsymbol{s}_l]) \right] \qquad V^\pi_{\text{soft}}(\boldsymbol{s}_t) = \alpha \log \int_{\mathcal{A}} \exp(\frac{1}{\alpha} Q^\pi_{\text{soft}}(\boldsymbol{s}, \boldsymbol{a})) d\boldsymbol{a}$$

$$\downarrow$$

$$Q^\pi_{\text{soft}}(\boldsymbol{s}_t, \boldsymbol{a}_t) = r_t + \gamma \mathbb{E}_{(\boldsymbol{s}_{t+1}, \ldots) \sim \rho_\pi} [V^\pi_{\text{soft}}(\boldsymbol{s}_{t+1})]$$

最終的な目的関数

$$\begin{aligned} J_{\text{maxent}}(\pi) &= \sum_{t=0}^{\infty} \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t) \sim \rho_\pi} [Q^\pi_{\text{soft}}(\boldsymbol{s}, \boldsymbol{a}) + \alpha \mathcal{H}_\pi[\boldsymbol{a}_t | \boldsymbol{s}_t]] \\ &= \sum_{t=0}^{\infty} \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t) \sim \rho_\pi} [r_t + \alpha \mathcal{H}_\pi[\boldsymbol{a}_t | \boldsymbol{s}_t] + \gamma V^\pi_{\text{soft}}(\boldsymbol{s}_{t+1})] \end{aligned}$$

SAPO

- SHACの枠組みにエントロピー最大化を加える
  - H stepのリターン

$$R^{\alpha}_{0:H}(\tau) = \sum_{t=0}^{H-1} \gamma^t (r_t + \alpha \mathcal{H}_\pi[\boldsymbol{a}_t|\boldsymbol{s}_t])$$

  - 1次推定量を利用した勾配

$$\hat{\nabla}^{[1]}_\theta J_{\mathrm{maxent}}(\pi) = \nabla_\theta (R^{\alpha}_{0:H} + \gamma^H V_{\mathrm{soft}}(\boldsymbol{s}_H))$$

  - 価値関数

$$\tilde{V}_{\mathrm{soft}}(\boldsymbol{s}_t) = \Gamma^{\lambda}_{t:t+H} \qquad \Gamma_{t:t+k} = \left(\sum_{l=0}^{k-1} \gamma^l (r_{t+l} + \alpha \mathcal{H}_\pi[\boldsymbol{a}_{t+l}|\boldsymbol{s}_{t+l}]) \right) + \gamma^k V_{\mathrm{soft}}(\boldsymbol{s}_{t+k})$$

  - 価値関数は以下の式を最小化するように学習

$$\mathcal{L}(V_{\mathrm{soft}}) = \sum_{l=t}^{t+H-1} \mathbb{E}_{s_l \sim \rho_\pi} \left[\left\| V_{\mathrm{soft}}(\boldsymbol{s}_l) - \tilde{V}_{\mathrm{soft}}(\boldsymbol{s}_l) \right\|^2 \right]$$

設計選択① エントロピーの利用

- αの自動チューニング
  - Lagrange Dualにより学習中にαを最適化

$$\min_{\alpha_t \geq 0} \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t) \sim \rho_\pi} [\alpha_t (\mathcal{H}_\pi [\boldsymbol{a}_t | \boldsymbol{s}_t] - \bar{\mathcal{H}})]$$

  - 目標エントロピー $\bar{\mathcal{H}} = -\dim(\mathcal{A})/2$ に近づくように更新
  - これにより、異なるタスクの異なる行動次元でも手動でのチューニングが不要

- エントロピー正規化
  - エントロピーを [0, +1]に正規化し、報酬依存性を低減

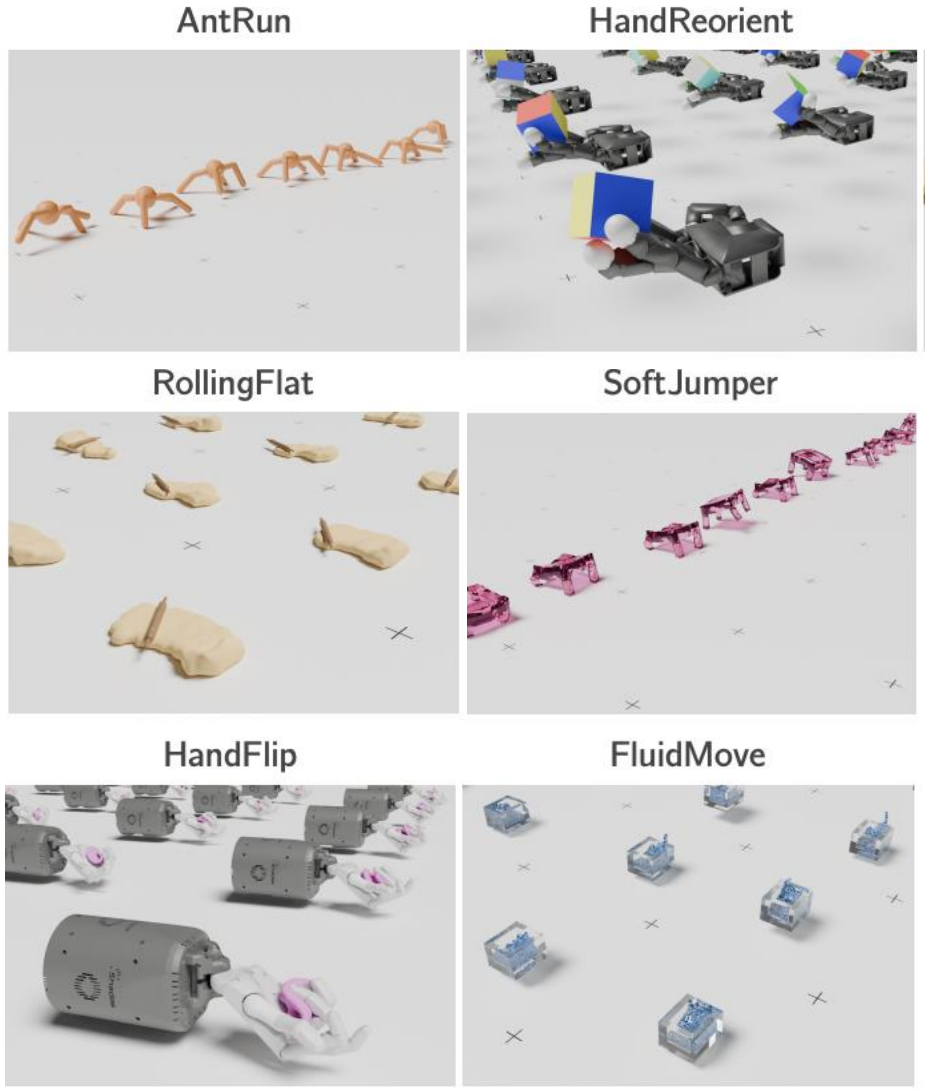
設計選択② SHACとの違い

- 状態依存分散
  - Actorは平均μのみでなく分散σ^2 も出力（SHACはμのみ）
  - 不確実ならσ↑で探索、$\pi_\theta = \tanh(\mathcal{N}(\mu_\theta(\mathbf{s}), \sigma_\theta^2(\mathbf{s})))$.
- Critic ensemble, no target networks
  - clipped double critic
    - 2つのCriticを学習、TD targetを作るとき二つの最小値を用いる（worst caseを見積り、過大評価を防ぐ）
    - Actor更新時には平均を利用し、学習スピードは保つ
  - Target networkは使用しない
- 最適化安定化
  - 活性化関数：ELU → SiLUを使用
  - 最適化手法：Adam → AdamW
  - 勾配クリッピング：1.0 → 0.5

実験　タスク設定

- 剛体タスク
  - AntRun：4脚の剛体の歩行
  - HandReorient：立方体を連続姿勢制御

- 変形体タスク
  - Rolling Flat：粘土をめん棒でならす
  - SoftJumper：弾性4足 ジャンプ移動
  - HandFlip：粘土を手のひらで半回転
  - FluidMove：コップ内の流体をこぼさず移送



実験　共通条件

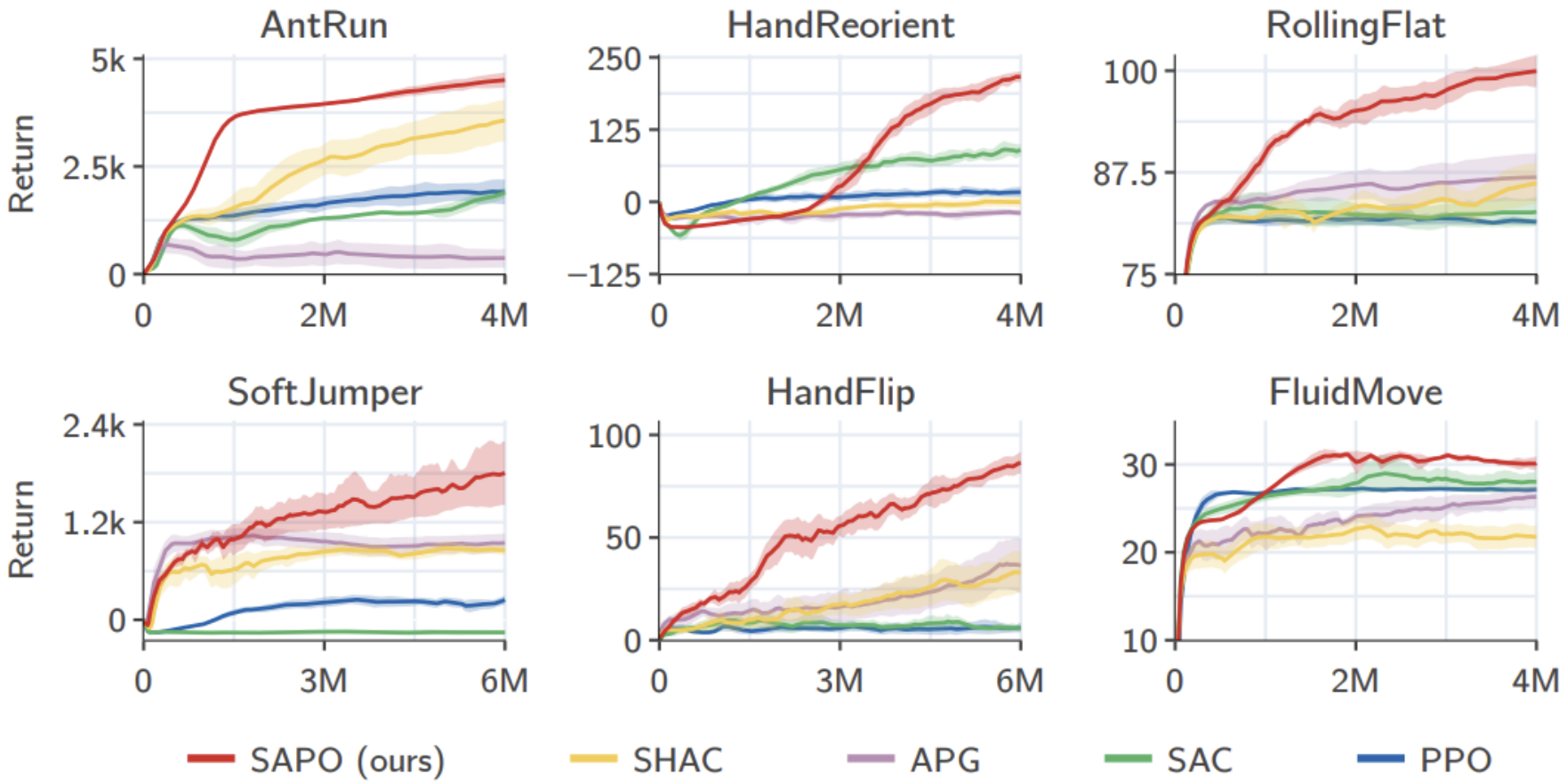
- 並列環境数　N=32 or 64
- エピソード長　T=200-400 step
- 総学習ステップ 4M-6M
- ベースライン
  - モデルフリー：PPO・SAC
  - FO-MBRL：APG・SHAC
  - TrajOpt：開ループ最適化

実験結果　定量結果

	AntRun	HandReorient	RollingFlat	SoftJumper	HandFlip	FluidMove
PPO	2048.7 ± 36.6	5.9 ± 4.9	81.2 ± 0.1	261.5 ± 12.4	7.3 ± 1.1	27.3 ± 0.2
SAC	2063.6 ± 13.9	70.5 ± 10.2	83.0 ± 0.3	-161.8 ± 2.5	4.6 ± 1.1	28.2 ± 0.7
TrajOpt	915.5 ± 29.6	-12.5 ± 2.0	81.5 ± 0.1	437.2 ± 17.7	27.3 ± 2.6	27.0 ± 0.1
APG	258.7 ± 20.3	-11.6 ± 1.9	86.9 ± 0.4	956.6 ± 15.6	38.2 ± 3.5	26.3 ± 0.3
SHAC	3621.0 ± 54.4	-2.5 ± 1.8	86.8 ± 0.4	853.3 ± 10.2	32.7 ± 2.9	21.7 ± 0.4
SAPO (ours)	4535.9 ± 24.5	221.7 ± 9.5	100.4 ± 0.4	1820.5 ± 47.9	90.0 ± 2.2	30.6 ± 0.4

- 全てのタスクで提案手法（SAPO）が最も良い性能
- HandReorientの大幅改善が顕著

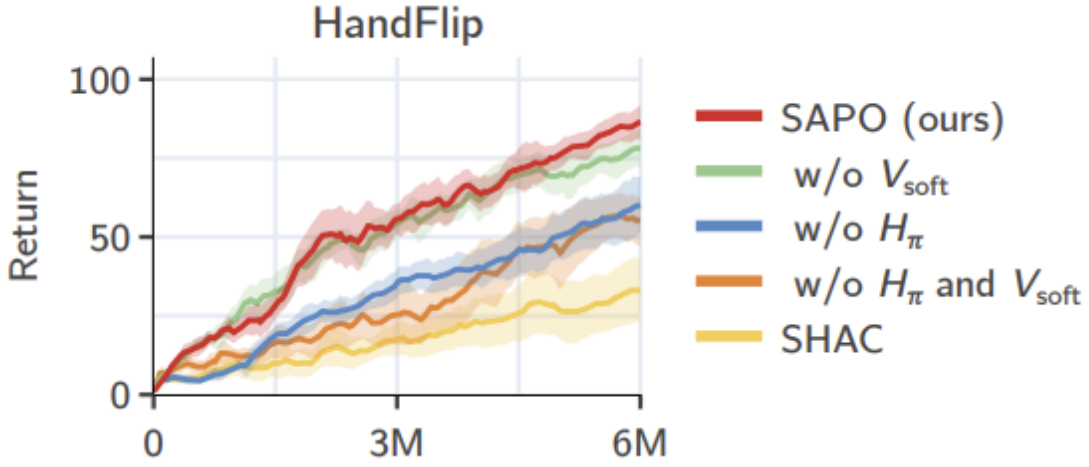
実験結果　学習曲線



- 他のFO-MBRLの手法に比べて、学習の安定性が高い

実験結果　アブレーション

- 検討パターン
  - 状態価値関数のエントロピー項を除いた場合
  - Returnのエントロピー項を除いた場合
  - 両方除いた場合



	HandFlip	(Δ%)
SAPO (ours)	90 ± 2	+172.7%
w/o V_{soft}	77 ± 3	+133.3%
w/o $\mathcal{H}_\pi$	59 ± 4	+78.8%
w/o $\mathcal{H}_\pi$ and V_{soft}	56 ± 3	+69.7%
SHAC	33 ± 3	–

- エントロピー項が最重要だった
- V_softによりターゲットの変動を抑制し、安定化

実験結果　ランドスケープ

Figure 12: **Loss surface comparison between algorithms – DFlex Ant.**

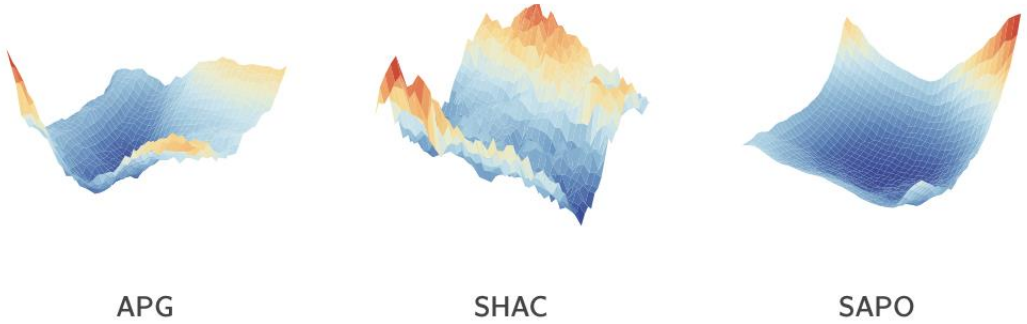


Figure 13: **Loss surface comparison between algorithms – Rewarped HandFlip.**

- SAPOでは他のアルゴリズムに比べてloss関数のランドスケープが滑らかに

考察と今後の方向性

- 観測問題
  - 今回は粒子状態を直接入力 → 実機では取得困難
- sim2realギャップ
  - マテリアルパラメータのずれ、センサノイズ
- 世界モデル × SAPO
  - Differentiable Simulatorを教師データとしてLatent Dynamicsを学習
  - モデル予測でHorizon拡大

まとめ

- Rewarped
  - “並列 × 微分可能 × マルチフィジックス” を初めて実現

- SAPO
  - 解析勾配 + 最大エントロピー
    - サンプル効率の向上
    - 変形体でも安定学習
    - 全てのタスクで最も良い性能