修士論文

ロボット制御の Sim-to-Real 転移のための環境乱択化を用いた正準動的モデルの学習

山之口 智也
奈良先端科学技術大学院大学
先端科学技術研究科
情報理工学プログラム

主指導教員: 杉本 謙二 教授
ロボットラーニング 研究室（情報科学領域）

令和 3 年 3 月 17 日提出

本論文は奈良先端科学技術大学院大学先端科学技術研究科に
修士 (工学) 授与の要件として提出した修士論文である。

山之口 智也

審査委員：

杉本 謙二 教授	（主指導教員）
池田 和司 教授	（副指導教員）
松原 崇充 特任准教授	（副指導教員）

ロボット制御の Sim-to-Real 転移のための環境乱択化を用いた正準動的モデルの学習*

山之口 智也

内容梗概

実世界におけるロボットの行動学習では，ロボットへの負荷や周囲の環境への負荷，そしてデータ収集にかかる時間が大きな問題となる．このような実世界でのロボット学習の問題を解決する方法として，シミュレーション環境で学習した方策を実機環境に転移可能にする Sim-to-Real が注目されている．しかし，既存の手法では全ての種類の方策に対する Sim-to-Real の転移技術を提供できておらず，画像入力を扱う方策の場合にはモデルフリー方策に限られる．しかし，ロボット制御においては，環境モデルのプランニングに基づくモデルベース方策が，タスクの転用性やデータ効率の点においてモデルフリー方策よりも優れていることが広く知られている．従って，画像入力を扱うモデルベース方策の Sim-to-Real 転移が実現できれば，高コストな実世界での試行錯誤を必要とすることなく，タスクの転用性に優れた方策を用いて実世界でロボット制御が可能となる．しかし，このようなモデルベース方策を実現するには「プランニングに要する計算量低減」「画像 reality gap への対応」「制御目的の設計容易」の 3 つを同時に兼ね備えたモデルが必要となるため，未だ実現には至っていない．本研究では，このような性質を満たす動的モデルの学習手法を提案する．提案手法の有効性を確認するため，バルブ押し込みタスクのシミュレーションを実施し，ベースラインとの比較実験を行った．

キーワード

Sim-to-Real，環境乱択化，深層学習，モデル予測制御，カルマン変分オートエンコーダ

*奈良先端科学技術大学院大学 先端科学技術研究科 修士論文, 令和 3 年 3 月 17 日.

Learning Canonical Dynamics using Domain Randomization for Sim-to-Real Transfer of Robotic Control *

Tomoya Yamanokuchi

Abstract

In learning robot behavior in the real world, the load on the robot, the load on the surrounding environment, and the time required for data collection are major problems. Sim-to-Real, which enables the transfer of policy learned in a simulation to a real world, has been attracting attention as a method to solve these problems. However, existing methods can not transfer all types of policy, and are limited to model-free policy in the case of policy that handle image input. On the other hand, it is widely known that model-based policy are superior to model-free policy in terms of task transferability and data efficiency. Therefore, if a Sim-to-Real transfer of model-based policy for handling image input can be realized, robotic control can be performed in the real world using policy with excellent task transferability without the need for costly trial-and-error in the real world. However, such a model-based policy has not yet been realized because it requires a model that 1) reduction of computational cost for planning, 2) reducing the image reality gap, and 3) ease of design for control objective. In this study, we propose a learning method of dynamics model that satisfies these properties. In order to confirm the effectiveness of the proposed method, we simulated a valve pushing task and compared it with a baseline.

Keywords:

Sim-to-Real, Domain Randomization, Deep Learning, Model Predictive Control , Kalman Variational Auto-Encoder

*Master's Thesis, Graduate School of Science and Technology, Nara Institute of Science and Technology, March 17, 2021.

Contents

List of Figures

List of Tables

1. 緒言

1.1 背景

ロボットが複雑な環境下において適切な行動を取るためには，自身の置かれている環境状態を適切に認識することが必要である．例えば物体を把持するだけでも，把持対象の物体がどのような形状をしていて，かつどのような姿勢をとっているかといった情報が必要になる．このような物体認識の技術は，近年の深層学習の発展に伴い飛躍的に進歩しており，単純な物体の位置認識から，2 次元画像からの 3 次元状態復元のような高度なものまで様々である．

このような深層学習技術の恩恵を受けるため，現在ではロボットの行動学習の研究においても深層学習が盛んに取り入れられている．しかし一方で，深層学習の技術を取り入れることはロボットの行動学習において利点ばかりではない．ロボットの行動学習を行う際，ロボットは環境から観測したデータを元に自身の行動を目的に合わせて最適化するが，観測データが画像のような高次元なデータである場合，学習に多くのデータを必要とする．そして，ロボットの行動学習におけるデータ収集は，ロボットが環境とのインタラクションを行うことによってのみ可能であるため，必要な学習データが増加することでロボットが行うべき試行錯誤の回数も増加する．しかし，実世界でロボットが多くの試行錯誤を行うことは，ロボットのハードウェアへの負荷の観点や，周囲の環境への負荷の観点から望ましくない．また，ロボットの限られた動作速度による実世界でのデータ収集には多くの時間を要するため，開発者への負担も大きい．

近年，このような実世界でのロボット学習における問題点を解決する方法として，シミュレーションを有効活用する方法が注目されている．シミュレーション環境の利点は，ロボットや物体を計算機上で仮想的に扱えるため，実世界の高価なロボットへのハードウェア負荷や周囲の環境に対する負荷を考えなくてよい点である．また，ロボットの実時間の動作速度に制限されることなくデータ収集が行える点や，環境内のすべての情報を取得できる点も大きな利点である．従って，シミュレーションを有効に活用することができれば，実世界でのロボットの行動学習における問題点の多くを解決することができると期待できる．ここで注意すべきは，ロボットのシミュレーションの技術自体は新しいものではなく，その活用方法の在り方が新しく注目されているという点である．実際に，現在でも新しいシミュレータは開発されているものの，古くから多くのシミュレータが存在していた．しかし，これまでのシミュレータの活用方法はあくまで実機ロボットで

実験を行う前の事前実験として位置づけられており，シミュレーション環境で獲得した知識を実機環境に引き継ぐような「知識の転用」は行われていなかった．このような知識の転用が行われてこなかった理由は，シミュレーション環境と実機環境の間に大きなギャップが存在するためだと考えられる．例えば，単純な位置制御においてさえ，目標位置を与えた際のロボットのアクチュエータダイナミクスはシミュレーション環境と実機環境で異なる．また，シミュレーション環境と実機環境での物体の摩擦係数が異なれば，物体とロボットがインタラクションする際のダイナミクスも異なる．またダイナミクスのギャップ以外にも，センサ観測に対する知覚のギャップが存在し，画像がその代表例として知られている．画像の知覚ギャップとは，画像からロボットや物体の状態を推定する状態推定器をシミュレーション環境で学習していたとしても，実機環境の画像に対しては同じような状態推定が行えない現象である．画像の知覚ギャップの例を図 1 に示す．近年注目されているシミュレーションの活用方法は，このようなシミュレーション環境と実機環境の間のギャップである「reality gap」を埋めるための技術であり,「Sim-to-Real」と呼ばれている．

Sim-to-Real の技術進歩により，現在では様々な reality gap に対処することが可能となった．しかし，画像入力を扱う既存の Sim-to-Real の手法は，タスクの転用性が低いモデルフリー方策に偏重している．モデルフリー方策とは，環境

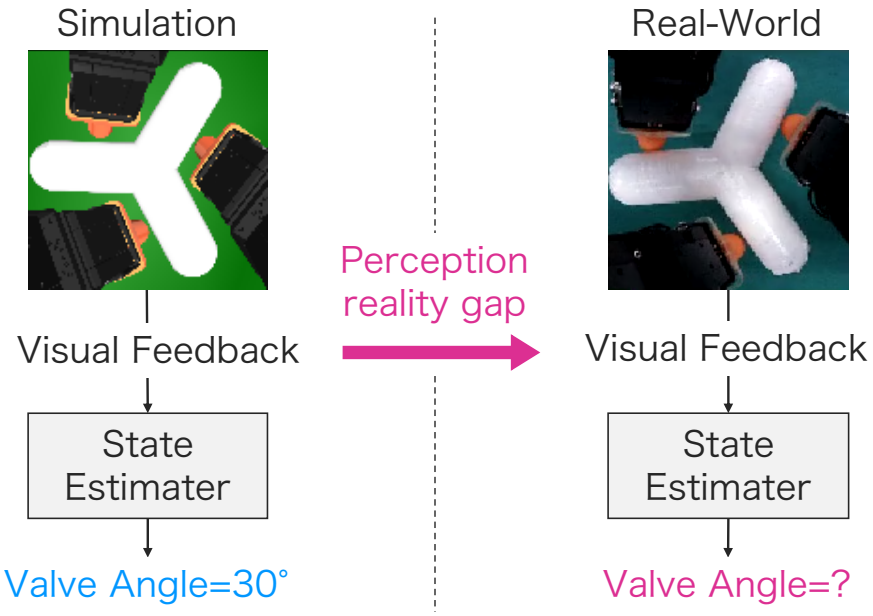


Fig. 1: Example of Perception reality gap

のモデル（制御入力に対する環境状態の時間発展を予測できるモデル）を持たない方策であり，プランニングを行うことなく少ない計算量で行動を出力することができる．しかし，モデルフリー方策では方策の学習時に制御目的を固定するため，後から制御目的を柔軟に変更することができない．ロボットアームを用いてある物体を目的の位置まで押す pushing タスクを例に考えれば，制御目的を物体の位置だけでなく，ロボットの速度も含めて考えたくなった場合，新しい制御目的に合わせて方策を変更することはできず，制御目的を変更した上で学習をやり直す必要がある．これに対し，モデルベース方策と呼ばれる方策では，方策が制御目的とは独立の環境のモデルを有するため，方策の学習後に柔軟に制御目的を変更することができる [17, 4]．しかし，モデルベース方策は行動を決定するためにプランニングを必要とするため，モデルフリー方策よりも計算量が多いという欠点がある．そして，プランニングの計算量は予測する状態の次元数に応じて増加するため，画像のような高次元データを扱う場合にはプランニングの計算量が深刻な問題となる．従って，画像入力を扱うモデルベース方策の Sim-to-Real を実現するためには，プランニングの計算量を抑えつつ，同時に画像入力に対する Sim-to-Real の学習を行えるようなモデルが必要となる．また，これらに加え，制御のしやすさの観点から考えれば制御目的が与えやすいモデルが必要となる．

1.2 目的

本研究の目的は,「現実的なプランニングの計算量」「実機画像に対応できる Sim-to-Real の機構」「制御目的の与えやすさ」の 3 つを同時に兼ね備えた，画像入力を扱うモデルベース方策を実現することである．提案手法の有効性を確認するため，ロボットハンドによるバルブ押し込みタスクにおいてベースラインとの比較実験を行った．

1.3 本論文の構成

本論文の残りの構成について説明する．2 章では提案手法と関わりの深い関連研究について述べる．3 章では，提案手法を理解する上での基礎知識となる事項について説明する．4 章では，提案手法として実画像に対応可能な画像の特徴抽出器と動的モデルを組み合わせたモデルについて説明する．5 章では，提案手法の有効性を確認するために行った 4 つのシミュレーション比較実験について説明する．6 章では，本論文のまとめと今後の展望について述べる．

2. 関連研究

シミュレーション環境と実機環境の間の reality gap を埋める Sim-to-Real の方法には様々な種類のものが存在するが，ここでは提案手法に深く関係する画像入力を扱う方法に限定して述べる．画像入力を扱う Sim-to-Real の方法は大きく 3 つあり，高品質なレンダリング画像を用いるアプローチ，ドメイン適応を用いるアプローチ，そして環境乱択化を用いるアプローチである．本章ではこれらの手法について説明する．

高品質なレンダリング画像を用いるアプローチ： シミュレーション環境と実機環境の reality gap を埋める最も単純な方法は，シミュレーション環境をなるべく実機環境に近づけることである．このような考えに基づき，James らによる研究 [11] では，実機環境に近い高品質なレンダリング画像を用いることで，方策が受け取る観測画像の reality gap を埋めることに挑戦しており，限られた条件下において実機ロボットアームによる箱の持ち上げタスクに成功している．しかし，あくまで限られた条件下での成功であるように，高品質なレンダリング画像を用いるアプローチでは，照明条件や物体表面のテクスチャ，カメラのセンサノイズといった様々な reality gap に対応することは困難であると考えられる．

ドメイン適応を用いるアプローチ： Bousmalis らの研究 [3] では，ソースドメインであるシミュレーション環境の画像を，ターゲットドメインである実機環境の画像へ写像するドメイン適応（Domain Adaptation: DA）を行うことで，シミュレーション環境の画像を実機環境の画像に変換した画像を生成している．これにより，方策はシミュレーション上で変換された画像を用いて学習できるため，物体把持タスクを成功させるために必要な実機環境での試行錯誤の回数を削減することに成功している．しかし，DA ではターゲットドメインとなる実機環境の画像が依然として必要であるため，事前に実機環境でデータ収集を行う必要がある．

環境乱択化を用いるアプローチ： James らの研究 [10] では，シミュレーション環境内のロボットや物体の見た目をランダムに変更することで，方策が受け取る入力画像を多種多様にする環境乱択化（Domain Randomization: DR）[21, 20] を行っている．DR を行うことにより，方策は入力画像の見た目に対して汎化するため，実画像が入力された場合でもそれを一種のランダム化画像とみなして適切な行動をとることが可能となる．この学習方法は，「シミュレーション環境の

ばらつきが十分に大きければ，シミュレーション環境で学習した方策であっても追加学習を必要とせずに実機環境に汎化できる」という仮説に基づく．これにより，James らの研究 [10] では実機環境の画像を一切必要とすることなく，ロボットアームによる片付けタスクを達成している．

また，James らの別の研究 [12] では，DR を発展させた手法として Randomized-to-Canonical Adaptation Networks（RCAN）が提案されている．RCAN は方策の入力にランダム化した観測画像を直接与える方法 [10] とは異なり，ランダム化画像を特定のターゲットドメイン画像に変換する変換ネットワークを方策とは別に学習する．ただし，ターゲットドメインとなる画像は実機環境の画像ではなく，正準環境と呼ばれる特定のシミュレーション環境の画像である．RCAN は深層ニューラルネットワーク（Deep Neural Network: DNN）によって構成され，敵対的生成ネットワーク（Generative Adversarial Network: GAN）[7] の枠組みで学習される．一度学習された RCAN は様々な見た目の入力画像に対して汎化しているため，実機環境の画像が入力された場合でもそれを一種のランダム化画像とみなし，対応する正準画像へと変換することができる．そして，RCAN によって出力された変換画像をあらかじめ正準環境で学習しておいた方策に入力することで，実機画像から適切な行動を決定することができる．

また，DR を RGB 画像のランダム化に使用するのではなく，深度画像に適用させた手法も提案されている．Pashevich らの研究 [18] では，深度画像に対して印加するノイズを実機環境に合わせて最適化することで，少数の実機データから把持・積み上げ．配置といったマニピュレーションタスクを行うことに成功している．しかし，深度画像を用いるアプローチでは色情報が取得できないため，適用可能なタスクが限定されてしまう問題がある．

3. 準備

3.1 線形ガウス状態空間モデル

3.1.1 モデル

時刻 t における観測を $\mathbf{a}_t$，制御入力を $\mathbf{u}_t$，潜在状態を $\mathbf{z}_t$ としたとき，システムは次のような形で表されるとする．

$$\mathbf{z}_t = f(\mathbf{z}_{t-1}, \mathbf{u}_t, \boldsymbol{\epsilon}_t) \tag{1}$$

$$\mathbf{a}_t = g(\mathbf{z}_t, \boldsymbol{\delta}_t) \tag{2}$$

ここで，f は状態遷移モデル，g は観測モデル，$\boldsymbol{\epsilon}_t$ は時刻 t におけるプロセスノイズ，$\boldsymbol{\delta}_t$ は時刻 t における観測ノイズを表す．

線形ガウス状態空間モデル（LGSSM: Linear Gaussian State Space Model）[16] では，状態遷移モデル及び観測モデルをそれぞれ以下のようにモデル化する．

$$p_{\gamma_t}(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_t) = \mathcal{N}(\mathbf{z}_t \mid \mathbf{A}_t\mathbf{z}_{t-1} + \mathbf{B}_t\mathbf{u}_t, \mathbf{Q}) \tag{3}$$

$$p_{\gamma_t}(\mathbf{a}_t \mid \mathbf{z}_t) = \mathcal{N}(\mathbf{a}_t \mid \mathbf{C}_t\mathbf{z}_t, \mathbf{R}) \tag{4}$$

ここで，行列 $\gamma_t = [\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t]$ はそれぞれ時刻 t における状態遷移行列，制御行列，観測行列である．また，$\mathbf{Q}, \mathbf{R}$ はそれぞれプロセスノイズと観測ノイズの共分散行列を表す．

観測系列を $\mathbf{a} = [\mathbf{a}_1, \ldots, \mathbf{a}_T]$，制御入力系列を $\mathbf{u} = [\mathbf{u}_1, \ldots, \mathbf{u}_T]$，潜在状態系列を $\mathbf{z} = [\mathbf{z}_1, \ldots, \mathbf{z}_T]$ とし，時刻 $t = 1$ における初期の潜在状態を $\mathbf{z}_1 \sim \mathcal{N}(\mathbf{z}_1|\mathbf{0}, \boldsymbol{\Sigma})$ とすれば，LGSSM の同時確率分布 $p(\mathbf{a}, \mathbf{z} \mid \mathbf{u})$ は式 (5) のように表される．

$$\begin{aligned} p_\gamma(\mathbf{a}, \mathbf{z} \mid \mathbf{u}) &= p_\gamma(\mathbf{a} \mid \mathbf{z}) p_\gamma(\mathbf{z} \mid \mathbf{u}) \\ &= \prod_{t=1}^{T} p_{\gamma_t}(\mathbf{a}_t \mid \mathbf{z}_t) \cdot p(\mathbf{z}_1) \prod_{t=2}^{T} p_{\gamma_t}(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_t) \end{aligned} \tag{5}$$

ここで，$\gamma = [\gamma_1, \ldots, \gamma_T]$ である．3.1.2 節，及び 3.1.3 節では表記の簡単化のため γ は省略する．

3.1.2 カルマンフィルタ

LGSSM はすべての変数がガウス分布で表現されるため，潜在状態の推論が厳密に計算できる．時刻 t における潜在状態 $\mathbf{z}_t$ を推論する際，過去から現在までの観測系列 $\mathbf{y}_{1:t}$，及び制御入力系列 $\mathbf{u}_{1:t}$ を用いる方法をカルマンフィルタを呼ぶ．

カルマンフィルタによる推論は予測ステップと更新ステップの 2 つのステップから成る．まず，予測ステップでは時刻 $t-1$ までに得られた観測系列と時刻 t までの制御入力系列を用いて潜在状態 $\mathbf{z}_t$ を次のように予測しておく．

$$p(\mathbf{z}_t \mid \mathbf{a}_{1:t-1}, \mathbf{u}_{1:t}) = \int \mathcal{N}(\mathbf{z}_t \mid \mathbf{A}_t \mathbf{z}_{t-1} + \mathbf{B}_t \mathbf{u}_t, \mathbf{Q}) \mathcal{N}(\mathbf{z}_{t-1} \mid \boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1}) \mathrm{d}\mathbf{z}_{t-1} \tag{6}$$

$$= \mathcal{N}(\mathbf{z}_t \mid \boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1}) \tag{7}$$

$$\boldsymbol{\mu}_{t|t-1} \triangleq \mathbf{A}_t \boldsymbol{\mu}_{t-1} + \mathbf{B}_t \mathbf{u}_t \tag{8}$$

$$\boldsymbol{\Sigma}_{t|t-1} \triangleq \mathbf{A}_t \boldsymbol{\Sigma}_{t-1} \mathbf{A}_t^\top + \mathbf{Q} \tag{9}$$

次に，更新ステップでは時刻 t で観測した $\mathbf{y}_t$ を用いて $\mathbf{z}_t$ を次のように求めることができる．

$$p(\mathbf{z}_t \mid \mathbf{a}_{1:t}, \mathbf{u}_t) = \mathcal{N}(\mathbf{z}_t \mid \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \tag{10}$$

$$\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t \mathbf{r}_t \tag{11}$$

$$\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{C}_t) \boldsymbol{\Sigma}_{t|t-1} \tag{12}$$

ここで，$\mathbf{r}_t$ は残差ベクトルと呼ばれ，予測した観測と実際に得られた観測との差によって次のように与えられる．

$$\mathbf{r}_t \triangleq \mathbf{y}_t - \hat{\mathbf{y}}_t \tag{13}$$

$$\hat{\mathbf{y}}_t \triangleq \mathbb{E}[\mathbf{y}_t \mid \mathbf{y}_{1:t-1}, \mathbf{u}_{1:t}] = \mathbf{C}_t \boldsymbol{\mu}_{t|t-1} \tag{14}$$

また，$\mathbf{K}_t$ はカルマンゲイン行列と呼ばれ，次のように与えられる．

$$\mathbf{K}_t \triangleq \boldsymbol{\Sigma}_{t|t-1} \mathbf{C}_t^\top \mathbf{S}_t^{-1} \tag{15}$$

$$\mathbf{S}_t \triangleq \mathrm{cov}[\mathbf{r}_t \mid \mathbf{y}_{1:t-1}, \mathbf{u}_{1:t}] \tag{16}$$

$$= \mathbb{E}[(\mathbf{C}_t \mathbf{z}_t + \boldsymbol{\delta}_t - \hat{\mathbf{y}}_t)(\mathbf{C}_t \mathbf{z}_t + \boldsymbol{\delta}_t - \hat{\mathbf{y}}_t)^\top \mid \mathbf{y}_{1:t-1}, \mathbf{u}_{1:t}] \tag{17}$$

$$= \mathbf{C}_t \boldsymbol{\Sigma}_{t|t-1} \mathbf{C}_t^\top + \mathbf{R} \tag{18}$$

ここで，$\boldsymbol{\delta}_t$ は $\mathcal{N}(\mathbf{0}, \mathbf{R})$ に従う観測ノイズである．

また，カルマンゲイン行列は逆行列補題を用いて式 (19) のように表すこともできる．

$$\mathbf{K}_t = \boldsymbol{\Sigma}_{t|t-1} \mathbf{C}_t^\top (\mathbf{C}_t \boldsymbol{\Sigma}_{t|t-1} \mathbf{C}_t^\top + \mathbf{R})^{-1} = (\boldsymbol{\Sigma}_{t|t-1}^{-1} + \mathbf{C}_t^\top \mathbf{R} \mathbf{C}_t)^{-1} \mathbf{C}_t^\top \mathbf{R}^{-1} \tag{19}$$

3.1.3 カルマンスムーザ

3.1.2 節で述べたカルマンフィルタは，時刻 t における潜在状態 $\mathbf{z}_t$ の推論に時刻 t までの観測系列 $\mathbf{y}_{1:t}$ を用いる方法であった．これに対し，時刻 T までの観測系列 $\mathbf{y}_{1:T}$ を用いる方法をカルマンスムーザと呼び，次のような式で表される．

$$p(\mathbf{z}_t \mid \mathbf{y}_{1:T}) = \mathcal{N}(\boldsymbol{\mu}_{t|T}, \boldsymbol{\Sigma}_{t|T}) \tag{20}$$

$$\boldsymbol{\mu}_{t|T} = \boldsymbol{\mu}_{t|t} + \mathbf{J}_t(\boldsymbol{\mu}_{t+1|T} - \boldsymbol{\mu}_{t+1|t}) \tag{21}$$

$$\boldsymbol{\Sigma}_{t|T} = \boldsymbol{\Sigma}_{t|t} + \mathbf{J}_t(\boldsymbol{\Sigma}_{t+1|T} - \boldsymbol{\Sigma}_{t+1|t})\mathbf{J}_t^\top \tag{22}$$

$$\mathbf{J}_t \triangleq \boldsymbol{\Sigma}_{t|t}\mathbf{A}_{t+1}^\top \boldsymbol{\Sigma}_{t+1|t}^{-1} \tag{23}$$

ここで，$\mathbf{J}_t$ は逆方向カルマンゲインと呼ばれる．

3.2 変分オートエンコーダ

今，データセットとして N 個の i.i.d サンプルから成る $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ を考える．そして，データ $\mathbf{x}_i$ は事前分布 $p_\theta(\mathbf{a})$ を持つ潜在変数 $\mathbf{a}$ より生成されるという確率的生成モデルを $p_\theta(\mathbf{x} \mid \mathbf{a})$ を考える．ここで θ は確率分布をモデル化するパラメータである．このとき，潜在変数 $\mathbf{a}$ の事後分布を求めるためにはベイズの法則に従い $p_\theta(\mathbf{a} \mid \mathbf{x}) = p_\theta(\mathbf{x} \mid \mathbf{a})p_\theta(\mathbf{a})/p_\theta(\mathbf{x})$ を計算する必要があるが，尤度関数にあたる $p_\theta(\mathbf{x} \mid \mathbf{a})$ をニューラルネットワークのような複雑な関数によって表現した場合，事後分布を厳密に計算することができない．これに対し，変分オートエンコーダ（VAE: Variational Auto-Encoder）[13] ではパラメータ ϕ を持つ近似事後分布 $q_\phi(\mathbf{a} \mid \mathbf{x})$ を導入し，生成モデル $p_\theta(\mathbf{x} \mid \mathbf{a})$ のパラメータ θ と同時に学習を行う．一般的に，$q_\phi(\mathbf{a} \mid \mathbf{x})$ はエンコーダ，$p_\theta(\mathbf{x} \mid \mathbf{a})$ はデコーダと呼ばれる．

データセットの対数周辺尤度は各データ点 $\mathbf{x}_i$ を用いて $\log p_\theta(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \sum_{i=1}^N \log p_\theta(\mathbf{x}_i)$ と表すことができ，各データ点についての対数周辺尤度は

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x} \mid \mathbf{a})p_\theta(\mathbf{a})\mathrm{d}\mathbf{a} \tag{24}$$

$$= \log \int q_\phi(\mathbf{a} \mid \mathbf{x}) \frac{p_\theta(\mathbf{x} \mid \mathbf{a})p_\theta(\mathbf{a})}{q_\phi(\mathbf{a} \mid \mathbf{x})}\mathrm{d}\mathbf{a} \tag{25}$$

$$\tag{26}$$

と表すことができる．ここで Jensen の不等式より

$$\log \int p(x)f(x)\mathrm{d}x \geq \int p(x) \log f(x)\mathrm{d}x \tag{27}$$

という関係性を利用すれば

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x} \mid \mathbf{a}) p_\theta(\mathbf{a}) \mathrm{d}\mathbf{a} \tag{28}$$

$$= \log \int q_\phi(\mathbf{a} \mid \mathbf{x}) \frac{p_\theta(\mathbf{x} \mid \mathbf{a}) p_\theta(\mathbf{a})}{q_\phi(\mathbf{a} \mid \mathbf{x})} \mathrm{d}\mathbf{a} \tag{29}$$

$$\geq \int q_\phi(\mathbf{a} \mid \mathbf{x}) \log \frac{p_\theta(\mathbf{x} \mid \mathbf{a}) p_\theta(\mathbf{a})}{q_\phi(\mathbf{a} \mid \mathbf{x})} \mathrm{d}\mathbf{a} \tag{30}$$

$$= \int q_\phi(\mathbf{a} \mid \mathbf{x}) \log p_\theta(\mathbf{x} \mid \mathbf{a}) \mathrm{d}\mathbf{a} + \int q_\phi(\mathbf{a} \mid \mathbf{x}) \log \frac{p_\theta(\mathbf{a})}{q_\phi(\mathbf{a} \mid \mathbf{x})} \mathrm{d}\mathbf{a} \tag{31}$$

$$= \langle \log p_\theta(\mathbf{x} \mid \mathbf{a}) \rangle_{q_\phi(\mathbf{a}|\mathbf{x})} - \mathrm{KL}(q_\phi(\mathbf{a} \mid \mathbf{x}) || p_\theta(\mathbf{a})) \tag{32}$$

$$= \mathcal{L}(\theta, \phi) \tag{33}$$

として対数周辺尤度の変分下限 $\mathcal{L}(\theta, \phi)$ が得られる．ここで $\langle \cdot \rangle$ は期待値計算を表しており，$\mathrm{KL}(\cdot)$ は Kullback-Leibler divergence (KL-divergence) を表す．

式 (32) における 1 項目はデコーダによる復元誤差を表し，2 項目はエンコーダによって得られた潜在変数と潜在変数の事前分布との間の KL-divergence の意味での距離を表す．従って，VAE ではを変分下限を最適化することで，データをよく復元でき，かつ事前分布の制約を満たすような潜在空間が学習できる．

事前分布は一般的に $p_\theta(\mathbf{a}) = \mathcal{N}(\mathbf{a} \mid \mathbf{0}, \mathbf{I})$ と置き，近似事後分布は多変量正規分布として式 (34) のように置かれる．

$$q_\phi(\mathbf{a} \mid \mathbf{x}) = \mathcal{N}(\mathbf{a} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{34}$$

ここで，近似事後分布はニューラルネットワークによりモデル化されるため，平均 $\boldsymbol{\mu}$ と共分散行列 $\boldsymbol{\Sigma}$ はともにニューラルネットワークの出力となっている．共分散行列 $\boldsymbol{\Sigma}$ は通常対角行列としてモデル化される．従って，KL-divergence の項は

$$\mathrm{KL}(q_\phi(\mathbf{a} \mid \mathbf{x}) || p_\theta(\mathbf{a})) = \mathrm{KL}(\mathcal{N}(\mathbf{a} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) || \mathcal{N}(\mathbf{a} \mid \mathbf{0}, \mathbf{I})) \tag{35}$$

$$= \frac{1}{2}(\mathrm{Tr}(\boldsymbol{\Sigma}) + \boldsymbol{\mu}^\top \boldsymbol{\mu} - D - \log |\boldsymbol{\Sigma}|) \tag{36}$$

として計算することができる．ただし D は潜在変数 $\mathbf{a}$ の次元である．

4. 提案手法

4.1 概要

画像入力を扱うモデルベース方策の Sim-to-Real 転移を実現するためには，以下の 3 つを同時に兼ね備えたモデルが必要となる．

1. 現実的なプランニングの計算量（実時間制御のため）
2. 実機画像に対応できる Sim-to-Real の機構（Sim-to-Real のため）
3. 目標状態の与えやすさ（タスク転用性のため）

本研究では，このような条件を満たすモデルとして以下のようなモデルを考える．

1. 画像の特徴抽出器と動的モデルを Disentangled な構造で有し，かつそれらを観測データの対数周辺尤度に対する変分下限で同時に最適化
2. 画像の特徴抽出器を Domain Randomization で学習
3. 動的モデルの内部状態にはシミュレーション環境の真の内部状態を使用

まず 1 つ目については．画像入力を扱うモデルにおいてプランニングの計算量を現実的に抑えるためには，高次元な画像の空間での計算を避けることが重要である．従って，画像を自己回帰的に計算するようなモデルは不適切であり，ロボットの実時間制御には適さない．より理想的には，画像の低次元な特徴量を抽出し，抽出した低次元の特徴量を用いてプランニングの計算える Disentangled な構造が望ましい．このようなモデルを実現する単純な方法としては，画像から特徴抽出を行う特徴抽出器をはじめに学習しておき，その後特徴抽出器から得られた特徴量を用いて動的モデルを学習する方法が考えられる．しかし，画像の特徴抽出器と動的モデルを別々に学習させた場合，画像の特徴抽出の過程で生じる誤差を動的モデルに反映できないため，モデルの相補性を考慮することができない．従って，Disentangled な構造に加えて，一貫した基準に基づき画像の特徴抽出器と動的モデルを同時に学習できるモデルを考える．

2 つ目については，画像入力に対する Sim-to-Real を行うための方法である．画像入力のための Sim-to-Real の手法のうち，実機環境の画像を一切必要とせず，高い性能を有していることから，本研究では RCAN を参考とした画像の特徴抽出器を考える．

3 つ目は，Sim-to-Real における「環境内の全ての状態にアクセスできる」という利点から得られるものである．画像からシステムの動的モデルを学習する場合，一般的には動的モデルの内部状態は潜在変数として教師なし学習の枠組みで学習される．従って，学習して得られる動的モデルの内部状態は解釈可能な値の空間を持たず，直接目標状態を与えることはできない．そのため，画像として目標状態を与えるか，潜在状態に対応する報酬モデルを別途学習しておく必要がある [8]．しかし，画像として目標状態を与える場合，速度のような 1 枚の画像からは抽出することができない状態を目標状態を与えるには，複数枚の画像を用いる複雑な処理が必要になってしまう．また，潜在状態に対応する報酬モデルを別途学習する方法では，制御目的ごとに報酬モデルを学習する必要があるため，モデルベース方策が持つタスクの転用性の利点が損なわれてしまう．一方で，モデルの学習をシミュレーション環境で行う Sim-to-Real の問題設定では，環境内の全ての状態にアクセスできるというシミュレーション環境の利点を活用することで，動的モデルの内部状態を観測変数として扱うことができる．

以上の 3 つを合わせた提案手法のグラフィカルモデルを図 2 に示す．ただし，図 2 において t は時間インデックス，$\mathbf{x}_t^{\mathrm{ran}}$ はシミュレータのランダム化画像，$\mathbf{x}_t^{\mathrm{can}}$ はシミュレータの正準画像，$\mathbf{a}_t$ は V-RCAN のエンコーダ $q_\phi(\mathbf{a}_t|\mathbf{x}_t^{\mathrm{ran}})$ から得られる観測画像の潜在表現（かつ LGSSM にとっての疑似観測），$\mathbf{z}_t$ はシミュレータの内部状態，$\mathbf{u}_t$ は制御入力を表す．

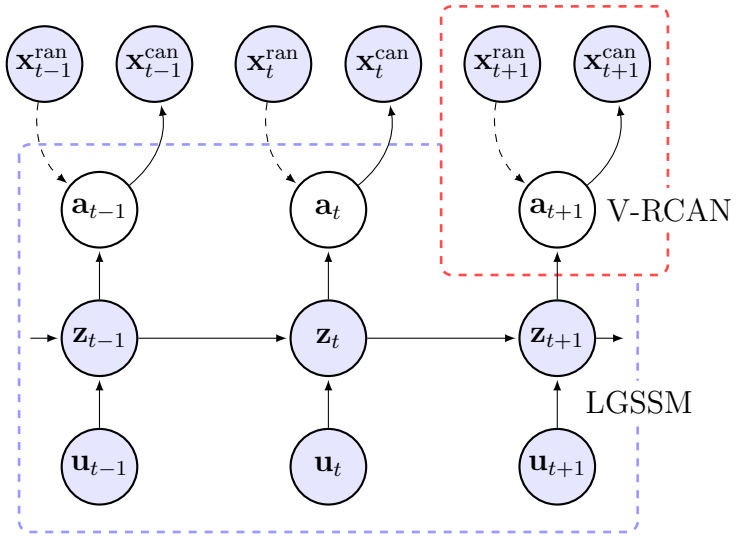


Fig. 2: Graphical Model of Proposed Method

4.2 正準画像系列に対する生成モデル

RCAN のネットワーク構造において，正準画像系列 $\mathbf{x}^{\mathrm{can}} = [\mathbf{x}_1^{\mathrm{can}}, .., \mathbf{x}_T^{\mathrm{can}}]$ は潜在変数 $\mathbf{a} = [\mathbf{a}_1, .., \mathbf{a}_T]$ から生成されたという確率的生成モデル

$$p_\theta(\mathbf{x}^{\mathrm{can}}|\mathbf{a}) = \prod_{t=1}^{T} p_\theta(\mathbf{x}_t^{\mathrm{can}}|\mathbf{a}_t) \tag{37}$$

を仮定する．ここで，$p_\theta(\mathbf{x}_t^{\mathrm{can}}|\mathbf{a}_t)$ はパラメータ θ を持つ DNN でモデル化する．同時に，$\mathbf{a}$ は LGSSM にとっての疑似観測としてモデル化する．この時，$\mathbf{a}$ の事前分布は $p_\gamma(\mathbf{a}|\mathbf{z}, \mathbf{u}) = p_\gamma(\mathbf{a}|\mathbf{z})$ となり，同時確率分布は式 (38) のように表される．

$$p(\mathbf{x}^{\mathrm{can}}, \mathbf{a}, \mathbf{z}|\mathbf{u}) = p_\theta(\mathbf{x}^{\mathrm{can}}|\mathbf{a})\, p_\gamma(\mathbf{a}|\mathbf{z})\, p_\gamma(\mathbf{z}|\mathbf{u}) \tag{38}$$

4.3 モデル学習アルゴリズムの導出

提案手法の学習則の導出は，カルマン変分オートエンコーダ（Kalman Variational Auto-Encoder: KVAE）[6] を参考に行う．潜在変数 $\mathbf{a}$ の事後分布は解析的に計算できないため，学習は観測データに対する対数周辺尤度

$$\mathcal{L} = \sum_{n}^{N} \log p_{\theta\gamma}(\mathbf{x}_{(n)}^{\mathrm{can}}, \mathbf{z}_{(n)}|\mathbf{u}_{(n)}) \tag{39}$$

の変分下限を最大化することで行う．ここで，N は系列数を表す．説明の簡略化のため，以後の説明では系列インデックス n は省略して表記する．次に，$\mathbf{a}$ の事後分布を近似するための変分分布をおく必要があるが，実機画像からも $\mathbf{a}$ を推定できる Sim-to-Real を行うため，ランダム化画像 $\mathbf{x}^{\mathrm{ran}}$ を用いて近似することを考える．従って，変分分布は $q(\mathbf{a}|\mathbf{x}^{\mathrm{ran}}, \mathbf{z}, \mathbf{u})$ と表わされ，対数周辺尤度の変分下限は

$$\log p(\mathbf{x}^{\mathrm{can}}, \mathbf{z}|\mathbf{u}) = \log \int p(\mathbf{x}^{\mathrm{can}}, \mathbf{a}, \mathbf{z}|\mathbf{u}) \mathrm{d}\mathbf{a} \tag{40}$$

$$\geq \left\langle \log \frac{p_\theta(\mathbf{x}^{\mathrm{can}}|\mathbf{a}) p_\gamma(\mathbf{a}|\mathbf{z}) p_\gamma(\mathbf{z}|\mathbf{u})}{q(\mathbf{a}|\mathbf{x}^{\mathrm{ran}}, \mathbf{z}, \mathbf{u})} \right\rangle_{q(\mathbf{a}|\mathbf{x}^{\mathrm{ran}}, \mathbf{z}, \mathbf{u})} \tag{41}$$

$$= \mathcal{F}(\theta, \gamma, \phi) \tag{42}$$

として得られる．ここで，ϕ は変分分布が持つパラメータである．変分分布は

$$q(\mathbf{a}|\mathbf{x}^{\mathrm{ran}}, \mathbf{z}, \mathbf{u}) = q_\phi(\mathbf{a}|\mathbf{x}^{\mathrm{ran}}) \tag{43}$$

とおき，DNN でモデル化する．これより，変分下限は改めて，

$$\mathcal{F}(\theta,\gamma,\phi)=\left\langle \log\frac{p_\theta(\mathbf{x}^{\mathrm{can}}|\mathbf{a})}{q_\phi(\mathbf{a}|\mathbf{x}^{\mathrm{ran}})}+\log p_\gamma(\mathbf{a}|\mathbf{z})\right\rangle_{q_\phi(\mathbf{a}|\mathbf{x})}+\log p_\gamma(\mathbf{z}|\mathbf{u}) \tag{44}$$

と書くことができる．この変分下限はサンプル $\{\tilde{\mathbf{a}}_i\}_{i=1}^I$ を用いたモンテカルロ積分により推定することができる．

4.4 動的パラメータネットワーク

LGSSM で非線形な動的モデルを学習する工夫として，KVAE を参考にした動的パラメータネットワークを導入する．本研究で用いる動的パラメータネットワークは，各時刻でそれまでの疑似観測 $\mathbf{a}_{0:t-1}$ と制御入力 $\mathbf{u}_{1:t}$ に依存して変化するパラメータ γ_t を出力するネットワークである．時刻 t での動的モデルの変化はシステムの過去の遷移と制御入力に依存すると考えられるため，γ_t は時刻 $t-1$ から時刻 t への動的モデルの変化を説明するようなパラメータとなっているべきである．従って，γ_t を $\mathbf{a}_{0:t-1}$ と $\mathbf{u}_{1:t}$ の関数として構成することで，$\mathbf{a}$ の事前分布は

$$p_\gamma(\mathbf{a}|\mathbf{z},\mathbf{u})=\prod_{t=1}^{T}p_{\gamma_t(\mathbf{a}_{0:t-1},\mathbf{u}_{1:t})}(\mathbf{a}_t|\mathbf{z}_t)\cdot p(\mathbf{z}_1)\prod_{t=2}^{T}p_{\gamma_t(\mathbf{a}_{0:t-1},\mathbf{u}_{1:t})}(\mathbf{z}_t|\mathbf{z}_{t-1},\mathbf{u}_t) \tag{45}$$

と書くことができる．動的パラメータネットワークは各時刻においてエンコードされた $\mathbf{a}_t$ と $\mathbf{u}_t$ を入力として受け取る LSTM（Long Short-Term Memory）と，LSTM の出力 $\mathbf{d}_t$ を受け取り softmax 変換を行う全結合層から成り，

$$\boldsymbol{\alpha}_t=\mathrm{softmax}(\mathbf{d}_t) \tag{46}$$

$$\mathbf{d}_t=\mathrm{LSTM}(\mathbf{a}_{t-1},\mathbf{u}_t,\mathbf{d}_{t-1}) \tag{47}$$

として構成される．従って，$\gamma_t=[\mathbf{A}_t,\mathbf{B}_t,\mathbf{C}_t]$ の各パラメータは以下のように表される．

$$\mathbf{A}_t=\sum_{k=1}^{K}\alpha_t^{(k)}(\mathbf{a}_{0:t-1},\mathbf{u}_{1:t})\mathbf{A}^{(k)} \tag{48}$$

$$\mathbf{B}_t=\sum_{k=1}^{K}\alpha_t^{(k)}(\mathbf{a}_{0:t-1},\mathbf{u}_{1:t})\mathbf{B}^{(k)} \tag{49}$$

$$\mathbf{C}_t=\sum_{k=1}^{K}\alpha_t^{(k)}(\mathbf{a}_{0:t-1},\mathbf{u}_{1:t})\mathbf{C}^{(k)} \tag{50}$$

このような，局所線形モデルのパラメータを観測データと制御入力を用いてモデル化するという考え方は，動的モデル学習における過去の研究 [2] においても用いられているため有効であると考えられる．

動的パラメータネットワークの構造を図示したものを図 3 に示す．

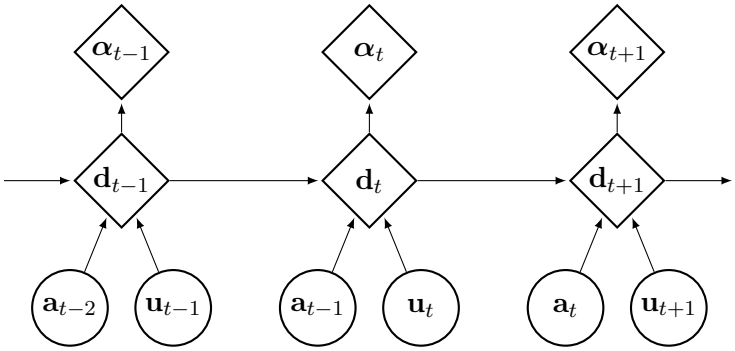


Fig. 3: Structure of Dynamics Parameter Network in Proposed Method

4.5 観測画像に基づくモデルの内部状態及び正準画像の推論

学習済みモデルを用いた推論では，毎時刻の観測画像 $\mathbf{x}_t$ から逐次的に潜在状態 $\mathbf{z}_t$ を更新するフィルタリングと，数ステップの観測を行った後に制御入力 $\mathbf{u}_{t:T}$ のみから潜在状態 $\mathbf{z}_{t:T}$ を生成する予測の 2 つがある．

フィルタリングでは，観測画像 $\mathbf{x}_t^{\mathrm{obs}}$ が得られたとき V-RCAN のエンコーダ $q_\phi(\mathbf{a}_t|\mathbf{x}_t^{\mathrm{obs}})$ から潜在表現 $\mathbf{a}_t$ をサンプリングし，これを用いて潜在状態の確率分布 $p_\gamma(\mathbf{z}_t|\mathbf{a}_t, \mathbf{a}_{1:t-1}, \mathbf{u}_{1:t})$ を式 (10)-(12) によって更新する．その後，潜在状態を画像に復元する場合には，更新した $\mathbf{z}_t$ の確率分布を用いて LGSSM の観測モデルである式 (4) より $\mathbf{a}_t$ を得て，最終的に $\mathbf{a}_t$ を V-RCAN のデコーダ $p_\theta(\mathbf{x}_t^{\mathrm{can}} \mid \mathbf{a}_t)$ で復元することで潜在状態 $\mathbf{z}_t$ に対応する正準画像 $\mathbf{x}_t^{\mathrm{can}}$ を得ることができる．

予測では，現在の潜在状態 $\mathbf{z}_t$ から LGSSM の状態遷移モデルである式 (3) を再帰的に適用することで，任意ステップ先の潜在状態が得られる．その後はフィルタリングと同様 V-RCAN のデコーダによって正準画像を復元することで，予測状態に対応する正準画像が得られる．

5. シミュレーション

5.1 概要

大きく分けて２つのシミュレーションを行った．１つ目は，提案手法における３つの工夫（計算速度のための Disentangled な構造，Sim-to-Real のための V-RCAN 構造，制御目的のための教師あり構造）がそれぞれ有効であるかどうかを確かめるため，それぞれの工夫がある場合とない場合での比較実験を行った．２つ目は，バルブの押し込みタスクにおいて提案手法の性能を評価した．全ての実験においてプランニングの方法には CEM を使用した．

5.2 環境

モデルの学習に使用するシミュレーションデータは Mujoco[22] を用いて収集した．シミュレーション環境には ROBEL D'Claw[1] 環境を用いた．D'Claw は 1 本あたり 3 自由度の指が 3 本で構成された合計 9 自由度のロボットハンドと，ロボットハンド下部に設置されたバルブで構成される．ただし本研究では，動的モデル学習の難しさを低減するため各関節は 3 本の指で全て同じ動きをするという制約を加え，３自由度の制御入力のみを持つものとした．また，各関節は全て可動域を [-30°, 30°] の範囲に設定し，制御方法には位置制御を用いた．

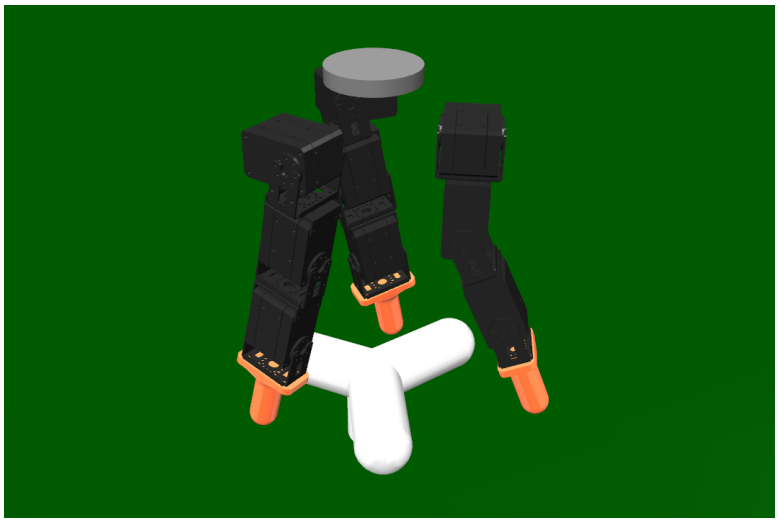


Fig. 4: Overview of ROBEL D'Claw Simulation Environment

5.3 シミュレーション 1：計算速度の比較

5.3.1 設定

画像入力を直接動的モデルの観測として使う場合と提案手法でモデルのプランニングに要する計算速度の比較を行った．比較モデルを図 5 に示す．この比較実験はあくまでプラニングに必要な計算速度を知ることが目的であるため，モデルのパラメータ学習は行っていない．従って，モデルのパラメータは全て初期化時の値をそのまま用いている．比較は表 1 に示す 3 つの方法で行った．

表 1 では各比較方法において変動させるパラメータを影付きセルとして表している．従って，方法 1 では CEM のサンプル数とホライゾンを固定させた状態で画像サイズを変化させた場合の計算速度を比較し，方法 2 では画像サイズとホライゾンを固定させた状態でサンプル数を変化させた場合の計算速度を比較し，方法 3 では画像サイズとサンプル数を固定させた状態でホライゾンを変化させた場合の計算速度を比較した．

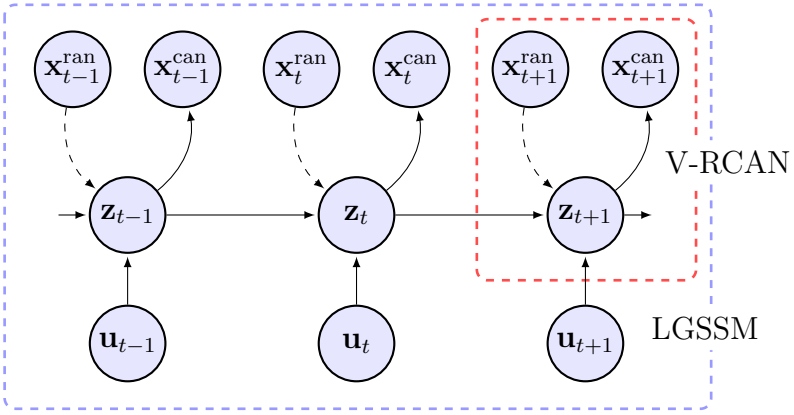


Fig. 5: Comparative Model for Comparing Computational Time

Table 1: Methods for Comparing Computational Time

Method	Image Size	Number of Samples	Horizon
Method1	$[10^0, 10^4]$	3000	15
Method2	$\{5, 64 \times 64 \times 3\}$	[1000, 5000]	15
Method3	$\{5, 64 \times 64 \times 3\}$	3000	[5, 30]

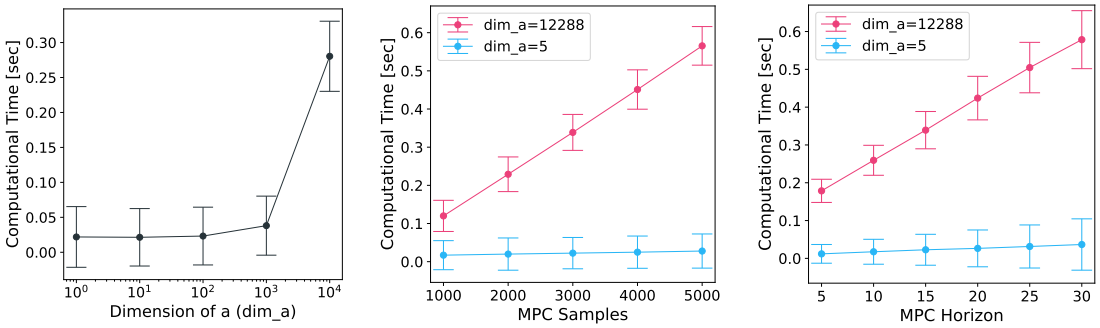
5.3.2 結果

計算速度の比較に関するシミュレーション結果を図 6 に示す．図 6a より，観測画像サイズが 10^4 程度まで増加すると計算時間が大きく増加することが確認でき，例えば 10^1 の場合と 10^4 の場合では計算速度に約 10 倍の差があることがわかる．ここで，$64 \times 64 \times 3$ という一般的な画像サイズの次元数は $12288 > 10^4$ であるため，観測画像を直接動的モデルの観測としてしまうと計算時間が大きく増加するポイントを超えてしまうことがわかる．また，これ以上大きな画像サイズ（例えば $128 \times 128 \times 3 = 49152$）の場合，一般的な GPU ではそもそも行列計算に必要なメモリ空間を確保できないという問題に直面する（実際に，TITAN RTX（24GB VARM）環境では $128 \times 128 \times 3$ の画像サイズは計算できないことを確認）．また，プランニングの計算が大きな画像サイズで仮にできたとしても，モデル学習の際にはさらに逆行列計算という 3 乗オーダーのより大きな計算が必要となるため，モデルの学習を行うという観点から考えても現実的ではない．

次に，図 6b の結果から，動的モデルの観測次元が 5 の小さい場合ではサンプル数の増加に伴う計算速度の増加が緩やかであるのに対し，観測次元が $64 \times 64 \times 3$ の場合ではサンプル数の増加に伴い計算速度も大きく増加していることがわかる．

また，図 6c の結果から，動的モデルの観測次元が 5 の小さい場合ではホライゾンの増加に伴う計算速度の増加が緩やかであるのに対し，観測次元が $64 \times 64 \times 3$ の場合ではホライゾンの増加に伴い計算速度も大きく増加していることがわかる．

従って，以上の 3 つの計算速度に関する比較結果より，高次元な画像を直接動的モデルの観測としてモデル化するべきではないと考えられる．



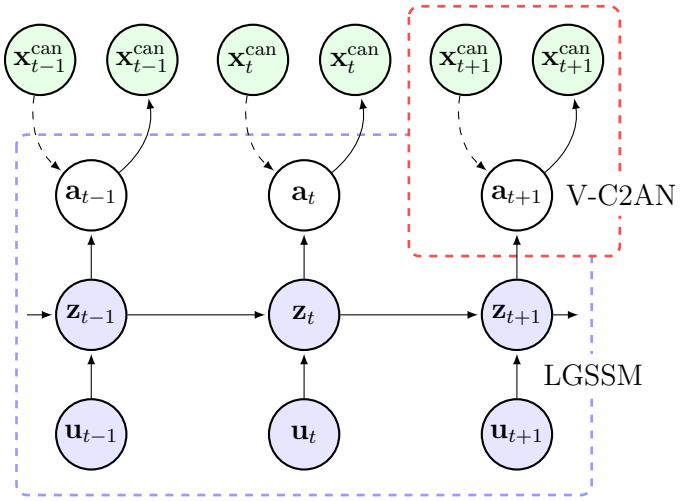
(a) Method1: Image Size (b) Method2: Number of Samples (c) Method3: Horizon

Fig. 6: Comparative Results of Computational Time

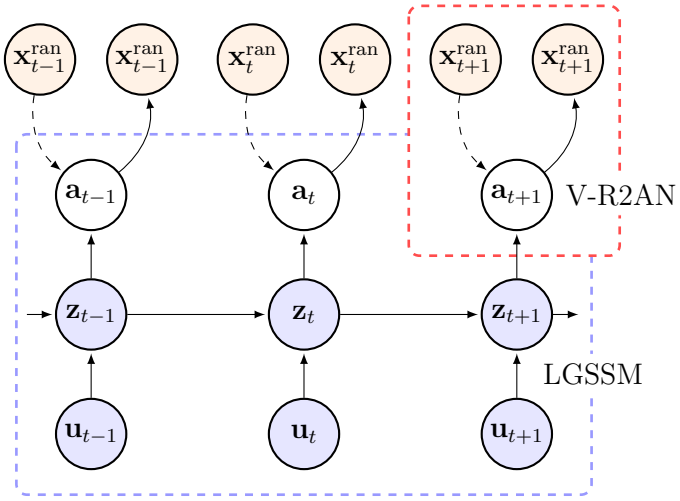
5.4 シミュレーション２：状態推定精度の比較

5.4.1 設定

画像の特徴抽出器の構成を変化させた２つの手法と提案手法とで，学習したモデルの状態推定精度の比較を行った．比較モデルを図７に示す．図７では比較手法と提案手法におけるモデルの違いをわかりやすくするため，正準画像 $\mathbf{x}^{\mathrm{can}}$ を緑の影付きノードで，ランダム化画像 $\mathbf{x}^{\mathrm{ran}}$ を黄色の影付きノードで表している．



(a) Comparative Model1: LGSSM + V-C2AN



(b) Comparative Model2: LGSSM + V-R2AN

Fig. 7: Comparative Models for Comparing State Estimation Error

比較手法 1 は画像の特徴抽出器が Randomized-to-Canonical（RC）の構造を取っておらず．Canonical-to-Canonical（C2）の構造になっている．従って提案手法が LGSSM+V-RCAN というモデルになっているのに対し，比較手法 1 は LGSSM+V-C2AN というモデルになっている．また，比較手法 2 は画像の特徴抽出器が Randomized-to-Randomized（R2）の構造を取っているため，LGSSM+V-R2AN というモデルになっている．正準画像から状態の推定を学習する比較手法 1 は正準環境という 1 つの環境にしか対応できないため，環境変化に対する状態推定精度の低下が最も大きと予想される．一方で，ランダム化画像から状態の推定を学習する比較手法 2 は比較手法 1 と比べると頑健であるとは予想されるものの，提案手法の状態推定精度には劣ると考えられる．

比較のため，D'Claw 環境でのバルブの押し込みタスクにおいてモデル学習用のデータ収集を行った．モデルの状態 $\mathbf{z}_t$ は指 1 本が持つ 3 関節の位置・速度，及びバルブの位置・速度合計 9 次元とし，制御入力は 3 関節の目標位置の 3 次元とした．1 系列 30 ステップとし，5000 系列のデータを収集した．このうち，2500 系列はガウス分布からサンプリングしたランダムな行動とし，残り 2500 系列は CEM を用いて収集した．CEM によるデータ収集では，ホライゾンを制御入力次元と 1 系列のステップ数の積（$3 \times 30 = 90$）とし，最終ステップにおけるバルブの回転角度が（正の値として）最も大きくなるようにコスト設計を行った．V-RCAN の詳細なネットワーク構造は付録 D に示す．

5.4.2 結果

状態推定精度の比較に関するシミュレーション結果を図 8 に示す．図 8 では横軸がモデルの学習エポック，縦軸がテストデータに対する状態推定誤差（カルマンフィルタの誤差）を表している．テストデータには学習データの中から抽出した 50 系列を用いており，カルマンフィルタを計算する際の観測画像にはランダム化画像を用いている．状態推定誤差の計算に用いた評価式を式 (51) に示す．

$$\mathrm{RMSE}(n) = \frac{1}{D}\sum_{d=1}^{D}\sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{z}_{t,d}^{\mathrm{true}} - \mathbf{z}_{t,d}^{\mathrm{filtered}}\right)^2} \tag{51}$$

ここで，T はステップ数，D は状態 $\mathbf{z}_t$ の次元数，$\mathbf{z}_{t,d}^{\mathrm{true}}$ は d 次元目の真の状態，$\mathbf{z}_{t,d}^{\mathrm{filtered}}$ は d 次元目のカルマンフィルタの結果から得られる状態，n は系列インデックスを表す．従って，式 (51) による計算される状態推定誤差とは「カルマンフィルタによるモデルの推定状態と真の状態との全ステップでの RMSE を状態の次

元で平均化した値」を意味する．そして，テストデータ 50 系列の各系列において式 (51) を計算し，その平均と標準偏差をプロットした結果が図 8 となる．

図 8 より，提案手法である LGSSM+V-RCAN の状態推定誤差が最も低いことがわかる．V-R2CAN ではランダム化画像が画像の特徴抽出器の入出力になっている自己符号化器の構造を取っているため，ランダムに変化する色情報を含めた画像を復元するように学習が行われる．しかし．色情報を復元しようとした場合，本来は同じ点に写像されるべき画像の組は潜在空間において異なる点に写像されることになり，動的モデルを学習する LGSSM にとって異なる観測として扱われてしまうという問題が発生する．従って，ランダム化画像から特徴を抽出しつつ，かつ抽出した特徴が色情報に依らないように学習を行う提案手法の状態推定精度が最も高いと考えられる．

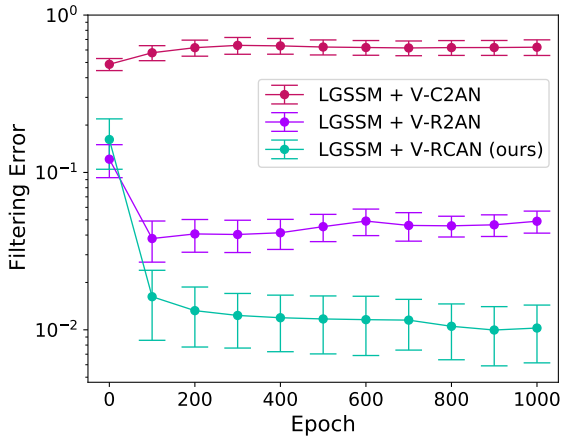


Fig. 8: Comparative Results of State Estimation Error

5.5 シミュレーション 3：状態予測精度の比較

5.5.1 設定

動的モデルの潜在状態 $\mathbf{z}_t$ を観測変数としない場合との比較を行った．比較モデルを図 9 に示す．動的モデルの潜在状態を観測変数として与えた場合，制御目的は与えやすくなるものの，観測画像と制御入力を説明するための潜在空間が強制的に固定されてしまう．しかし，固定された真の状態がモデルを学習する上で最適かどうかは不明であるため，真の状態を与えたことにより逆にモデルの学習性能が低下してしまう可能性も考えられる．従って，この比較実験では動的モデルの

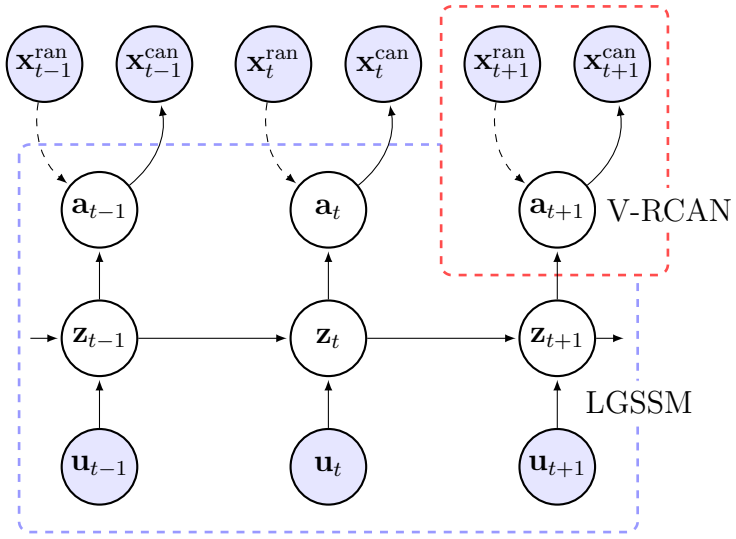


Fig. 9: Comparative Model for Comparing State Prediction Precision

潜在変数を真の状態に置き換えた場合に，著しい性能の低下が生じるかどうかを確認することが目的である．

モデルの学習に用いるデータには 5.4 節のシミュレーション実験と同様のものを用いた．モデルの性能評価は，テストデータ系列に対するモデルの予測画像と真の画像との類似度評価によって行った．画像の類似度評価手法には PSNR (Peak-Signal to Noise Ratio) ，及び SSIM (Structural Similarity)[24] を用いた．PSNR と SSIM は画像の予測タスクにおいても広く使用されている評価手法である [14, 5]．具体的には，まずはじめにモデルは初期観測を 10 ステップ行い潜在状態を推定し，その後 20 ステップ先までの各時刻における画像を制御入力のみから予測した．そして，予測した 20 ステップにおいてモデル予測画像と真の画像との間の PSNR 及び SSIM を計算し，20 ステップでの PSNR と SSIM の平均を計算した．状態予測精度の計算に用いた評価式を式 (52) と式 (53) に示す．

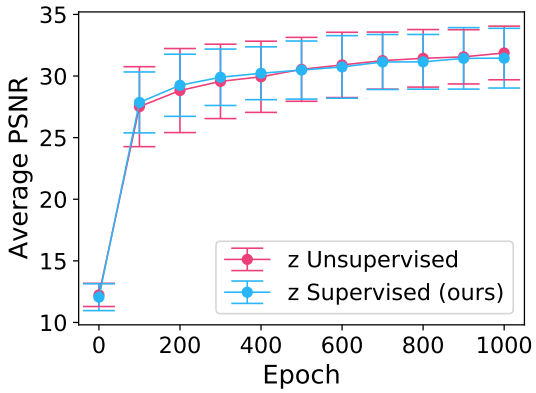
$$\text{Averange PSNR} = \frac{1}{20}\sum_{t=11}^{T=30} \text{PSNR}\left(\mathbf{x}^{*}_{t}, \mathbf{x}^{\text{predict}}_{t}\right) \tag{52}$$

$$\text{Averange SSIM} = \frac{1}{20}\sum_{t=11}^{T=30} \text{SSIM}\left(\mathbf{x}^{*}_{t}, \mathbf{x}^{\text{predict}}_{t}\right) \tag{53}$$

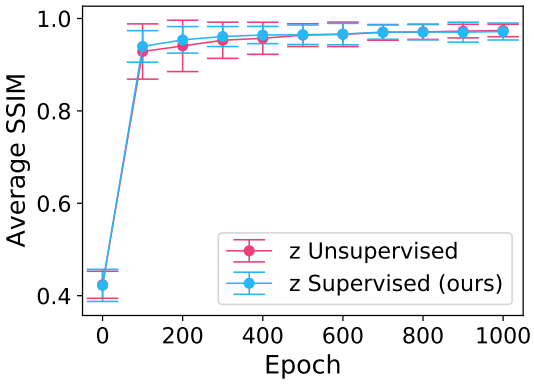
ここで，$\text{PSNR}(a, b)$ と $\text{SSIM}(a, b)$ は画像 a と b の間の PSNR 及び SSIM を表し，$\mathbf{x}^{*}_{t}$ と $\mathbf{x}^{\text{predict}}_{t}$ は時刻 t における真の画像とモデルの予測画像を表す．

5.5.2 結果

テストデータ 50 系列に対して式 (52) と式 (53) を適用し，50 系列での平均と分散を計算した結果を図 10 に示す．図 10 では，横軸が学習エポック数，縦軸が評価した PSNR と SSIM の値を表している．図 10 より，動的モデルの潜在状態を観測変数として与えた場合でも，著しい性能の低下は生じていないことが確認できる．また，提案手法の最終的な状態予測精度は比較手法にわずかに劣るものの，学習エポック数が 400 程度までの学習初期段階においては評価結果のばらつきが小さいことが確認できる．これは，学習初期においては真の状態がモデル学習のための有益な情報となっているのに対し，学習後期では細かなモデルの調整にとって真の状態が制約となってしまっているためだと考えられる．比較手法と提案手法における実際の予測画像の例を図 11 に示す．



(a) PSNR



(b) SSIM

Fig. 10: Comparative Results of Comparing State Prediction Precision

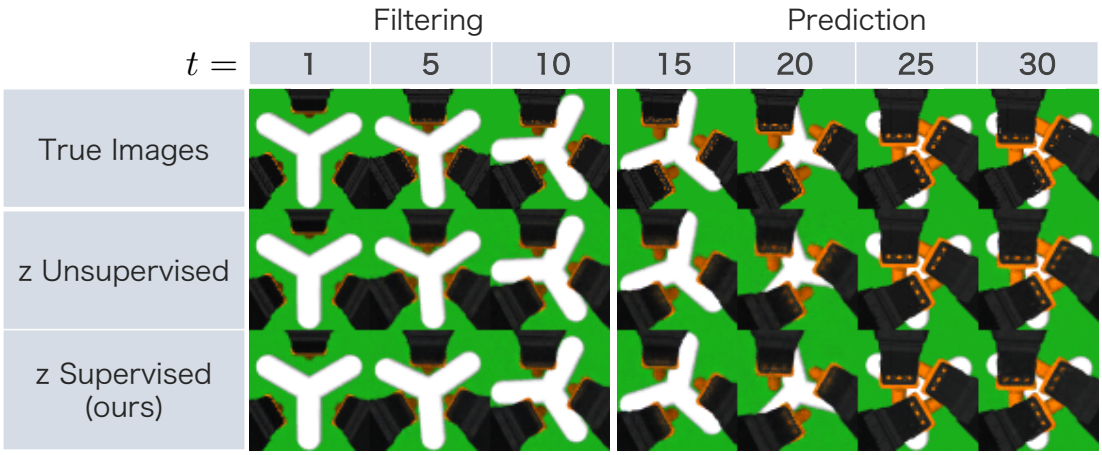


Fig. 11: Predicted Images for Test Data

5.6 シミュレーション 4：バルブ押し込みタスクにおける性能評価

5.6.1 設定

バルブ押し込みタスクにおけるタスク性能の評価を行った．モデルの学習データ及び比較手法には 5.4 章と同様のものを用いた．タスクのステップ数は 25 とし，各ステップにおける目標軌道と，目標軌道に対して求めた CEM の最適軌道系列との間の誤差をタスク性能として評価した．目標軌道には，学習データの中で最もバルブを押し込んだ系列を用いた．また，CEM の最適化では毎ステップごとの初期の制御入力系列のサンプリングにのみ範囲 $[-1.5, 1.5]$ の一様分布を用いた．実験における CEM の設定の詳細は付録 C.1 に示す．タスク性能の評価は図 12 に示すような 4 つの環境で行った．ここで，Env1 は正準環境，Env4 はランダム化環境に相当する．

(a) Env1 (b) Env2 (c) Env3 (d) Env4

Fig. 12: Images of Test Environments

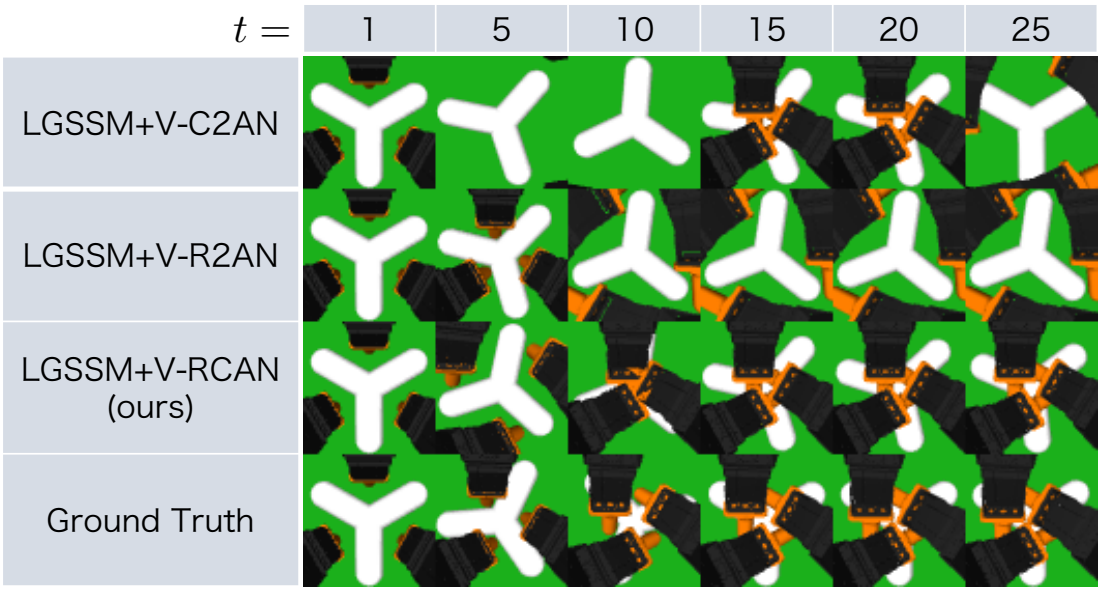
5.6.2 結果

タスク性能の比較結果を表2に示す．ただし，Env4ではランダム化画像の違いによる結果の影響を考慮するため，10個の異なるランダム化環境で評価を行い，平均と標準偏差を計算した．表2より，4つの環境のうち3つの環境において提案手法のタスク性能が最も高いことが確認できる．また，比較手法であるLGSSM+V-R2ANが最も高いタスク性能となっているEnv3においても，提案手法と比較手法の間のタスク性能の差は他の3つの環境の場合と比較して小さいことがわかる．

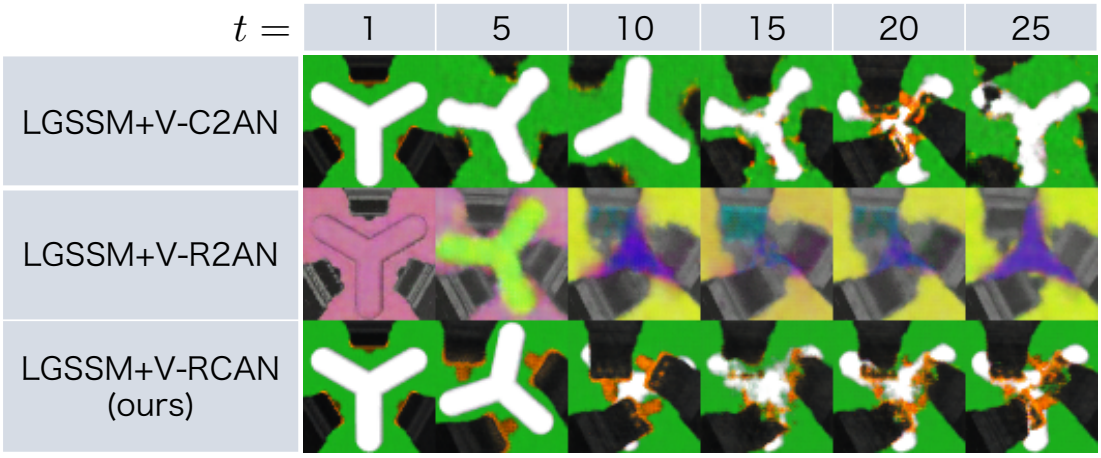
タスク実行時の実際のシミュレーション画像，及びモデルのフィルタリング画像の結果を図13から図16に示す．図14から図16の結果より，LGSSM+V-C2ANでは正準画像のみを用いた学習を行っているため，環境が変化した際にモデルの推定状態から復元した画像が真の画像と大きく異なることが確認できる．また，LGSSM+V-R2ANではランダム画像の色情報を復元するように学習が行われているため，モデルの推定状態から復元した画像の色情報が同一環境内でも大きく変化していることが確認できる．

Table 2: Result of Simulation 5.6

Method	Env1	Env2	Env3	Env4
LGSSM + V-C2AN	2.83 ± 1.96	1.85 ± 1.56	4.53 ± 1.66	3.96 ± 2.10
LGSSM + V-R2AN	4.11 ± 3.82	11.57 ± 8.48	$\mathbf{0.45 \pm 0.19}$	3.20 ± 4.36
LGSSM + V-RCAN (ours)	$\mathbf{1.66 \pm 1.34}$	$\mathbf{0.06 \pm 0.05}$	0.74 ± 1.49	$\mathbf{0.98 \pm 1.61}$

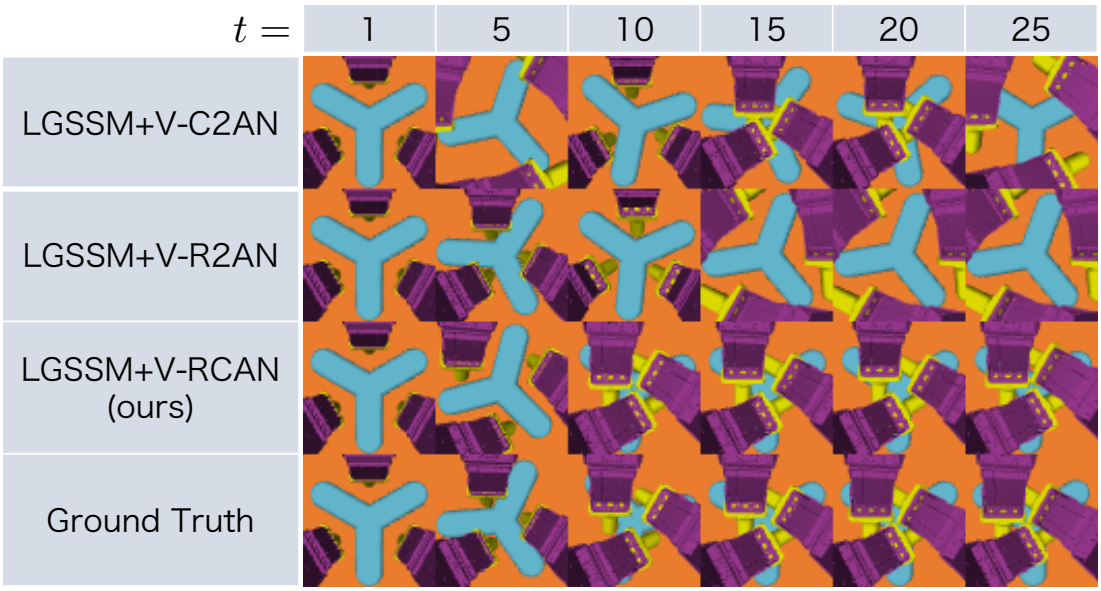


(a) Simulation Images

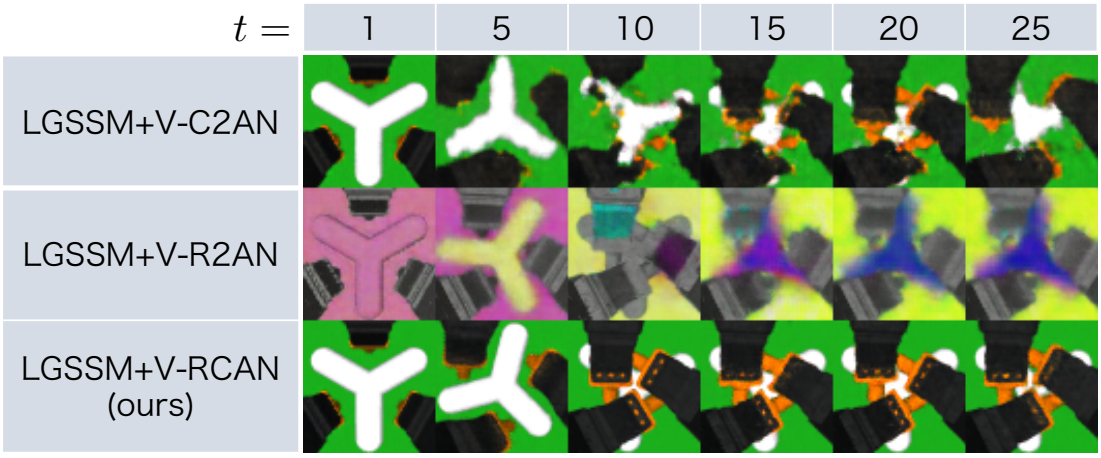


(b) Filtered Images

Fig. 13: Simulation and Filtered Images in Env1

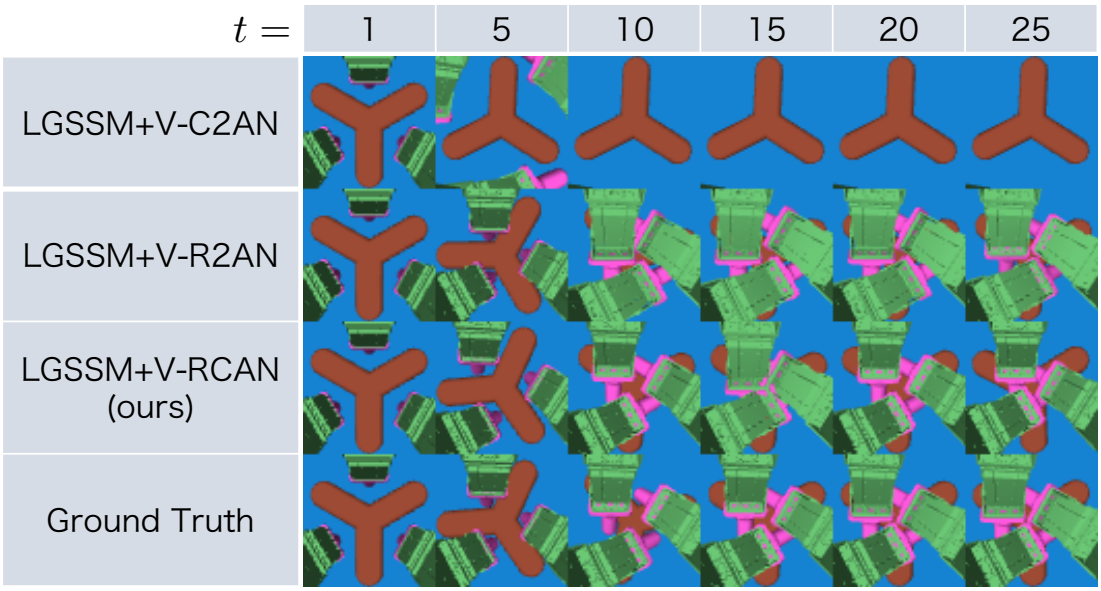


(a) Simulation Images

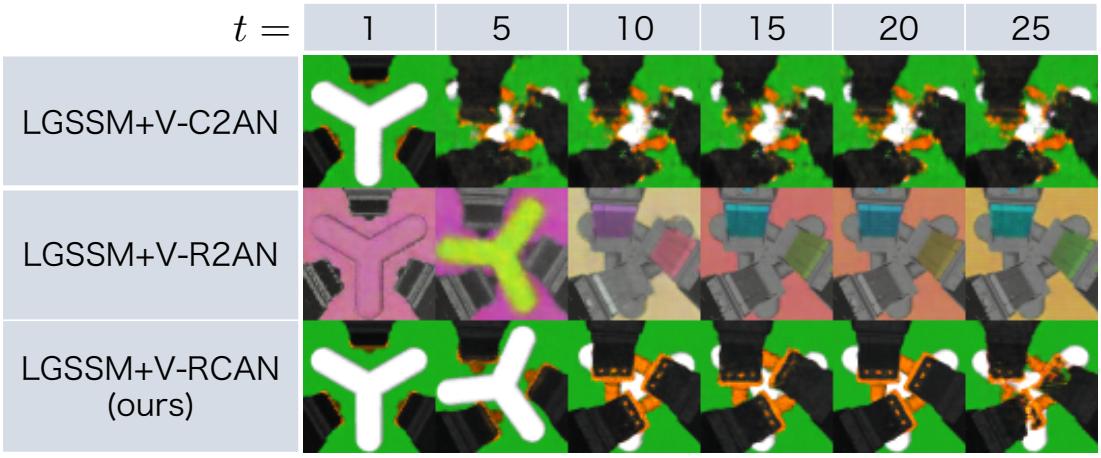


(b) Filtered Images

Fig. 14: Simulation and Filtered Images in Env2

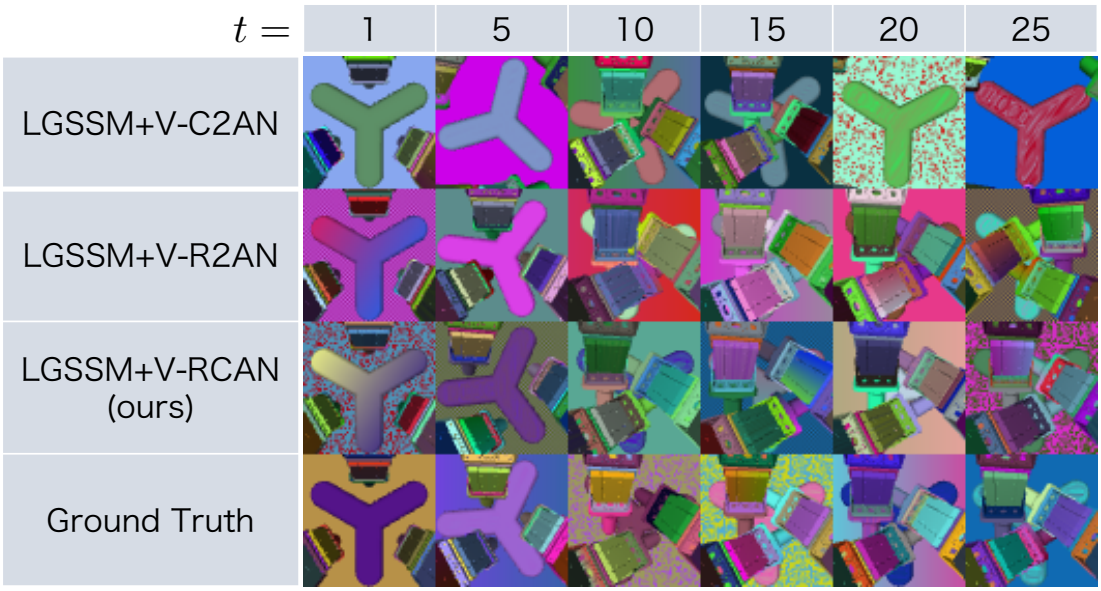


(a) Simulation Images

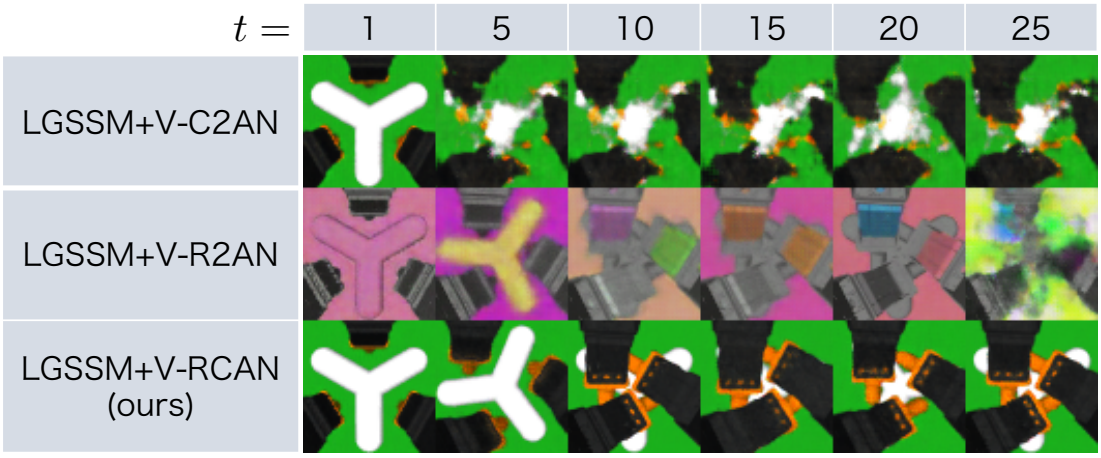


(b) Filtered Images

Fig. 15: Simulation and Filtered Images in Env3



(a) Simulation Images



(b) Filtered Images

Fig. 16: Simulation and Filtered Images in Env4

6. 結言

6.1 まとめ

本研究では,「現実的なプランニングの計算量」「実機画像に対応できる Sim-to-Real の機構」「制御目的の与えやすさ」の 3 つの性質を同時に兼ね備えた，画像入力を扱うモデルベース方策のための Sim-to-Real 転移手法を提案した．3 つの各性質を満たすため,「画像の特徴抽出器と動的モデルを Disentangled な構造で有し，かつそれらを観測データの対数周辺尤度に対する変分下限で同時に最適化」「画像の特徴抽出器を Domain Randomization で学習」「動的モデルの内部状態にはシミュレーション環境の真の内部状態を使用」という工夫をそれぞれで行った．提案手法の有効性を確認するため，シミュレーション環境でベースラインとの比較実験を行い，それぞれの工夫が有効であることを確認した．

6.2 今後の課題

実機実験： 提案手法の有効性を評価するためには，実機環境におけるタスク性能の評価が必要である．従って，シミュレーション環境と同様のロボットハンドによるバルブ押し込みタスクを実機環境で実施し，ベースラインとの比較実験を行うことを検討している．

タスク転用性の評価実験： 提案手法のモデルベース方策としての有効性を評価するためには，複数のタスクのもとでタスク性能を評価する，タスク転用性の評価実験が必要である．例えば，ロボットハンドによるバルブ押し込みタスクから派生するタスクとしては，完全にバルブを押し込むのではなく途中の角度で止めるタスクが考えられる．また別のタスクとしては，タスク実行時のロボットハンドのの関節速度に制約を加えた上でバルブを押し込むタスクも考えられる．

動的モデルの reality gap の考慮： 今回提案した手法では，Sim-to-Real のための工夫は観測画像に対する DR の学習のみである．しかし，1.1 章でも述べたとおり，実際には観測画像の reality gap 以外にも動的モデルの reality gap が存在する．従って，より精度の高い Sim-to-Real を実現するためには，動的モデルの reality gap に対処するための工夫が必要となる．動的モデルの reality gap を埋める方法としては，Peng らによる RDPG (Recurrent Deterministic Policy Gradient)[25]

を用いた方法 [19] が有名である．この方法では,「再帰型の方策が持つ内部状態に過去の状態行動系列の情報は全て含まれているため，内部状態から環境が持つ動的モデルを決定付けるパラメータを暗黙的に推定することができる」という仮説に基づき，シミュレーション環境で動的モデルに関係するダイナミクスパラメータをランダムに変化させながら LSTM で構成される方策を学習する．これにより，従来では必要であった推定すべきダイナミクスパラメータ集合の明示的な定義 [9] を避けながら，動的モデルの reality gap に対応することを可能にしている．一方で，提案手法で使用している動的パラメータネットワークは LSTM をによって構成されているため，Peng らの仮説に基づけば，ダイナミクスパラメータをランダムに変化させたシミュレーションデータを学習データに追加するだけで，そのまま動的モデルの reality gap に対応できるとも考えられる．

環境乱択化の強化： 観測画像の Sim-to-Real の精度を高めるためには，単純な画像の見た目の DR 以外にも，カメラの位置や姿勢もランダム化することが有効である．また，環境内への障害物の設置も有効であると考えられる．

能動的環境乱択化： シミュレーション環境内の状態をランダム化する DR では，各状態をどの程度の範囲でランダム化するかが重要となる．ランダム化する範囲が小さすぎる場合，reality gap に対応できるだけの汎化性能が得られず，逆にランダム化する範囲が大きすぎる場合，方策学習の難易度が無駄に高くなってしまう．この問題は，画像の見た目，カメラの位置，ダイナミクスパラメータといった様々なパラメータをランダム化する場合において非常に重要となる．従って，能動的環境乱択化 [15] のような方策学習にとって効率のよい DR を行うことが重要であると考えられる．

長期ステップタスクへの適用： 今回行ったシミュレーション実験では，タスクのステップ数は合計で 30 ステップであった．一方で，バルブを回転させ続けるようなより複数な作業を行うためには，長期ステップタスクを実行できる必要がある．しかし，長期ステップタスクではモデルのプランニングの際に蓄積される誤差が深刻になるため，長期ステップの状態予測を可能にする工夫を取り入れる必要があると考えられる．

謝辞

知能システム制御研究室の杉本謙二教授には，大学院入学前では学会において研究に関する貴重なご意見を頂き，大学院入学時には受験のための多くのご支援を頂き，大学院入学後は研究や講義，推薦書の執筆といった様々な面において多大なご支援を賜りました．心より感謝致します．

数理情報学研究室の池田和司教授には，ご多忙にもかかわらず副指導教員を引き受けていただき，成果報告会では研究に関する貴重なご指摘，ご意見を頂きました．深く感謝致します．

ロボットラーニング研究室の松原崇充特任准教授には，本研究の基礎となるロボットの行動学習に関する知識から，研究の進め方や論文執筆に関するご指導など，多くの貴重なご指導を頂きました．心より感謝致します．

ロボットラーニング研究室の秘書である西村啓美さん，林英子さん，藤本亜矢子さんには，学会に関する手続きや各種備品の管理など，日々の研究生活を送る上で欠かせない様々なご支援を頂きました．深く感謝致します．

知能システム制御研究室の小林泰介助教には，研究に関するご指摘やご意見を学内・学会問わず多く頂きました．心より感謝致します．

ロボットラーニング研究室の鶴峯義久特任助教には，本研究を進める上で重要となる様々な知識やご指導を頂き，また，研究に関する相談をした際には常に親切かつ丁寧にサポートして頂きました．深く感謝致します．

ロボットラーニング研究室博士後期課程 2 年の佐々木光さんには，ロボットを制御するための様々な知識から，ガウス過程回帰を用いた方策探索に関する知識など，多岐に渡る知識を教えて頂きました．心より感謝致します．

ロボットラーニング研究室博士後期課程 1 年の権裕煥さんには，本研究で扱う KVAE に関して，その基礎知識からコーディングに渡るまで，細かなご指導をいただきました．深く感謝致します．

また，一年先に御卒業されたロボットラーニング研究室の先輩である松岡潤樹さんと宮本知弥さんには，一番身近な先輩として，入学当初の右も左もわからない状態から生活面・授業面・研究面などの多くの面において親身に支えていただいたことを心より感謝致します．

ロボットラーニング研究室博士前期課程 2 年の同期である Oh Hanbit さん，角川勇貴さん，米野尚斗さんとは，入学当初の不安と希望で一杯の状態から，日々の生活・講義・研究を共にしてきた．時には授業の課題と研究に挟まれ，夜を徹して作業したこともあったが，互いに励まし合い，支え合うことで困難を乗り越

えてきた．心より感謝の意を表する．

また，日々の研究活動において様々なご意見，ご指摘等をくださったロボットラーニング研究室の皆様に，心より感謝致します．

最後に，大学院への進学を快く認めて頂き，日々の生活を経済面・精神面などから支えて頂いた両親に心より感謝致します．

参考文献

[1] 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 (CoRL)*, pages 1300–1313, 2019.

[2] Ershad Banijamali, Rui Shu, Mohammad Ghavamzadeh, Hung Bui, and Ali Ghodsi. Robust Locally-Linear Controllable Embedding. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 84, pages 1751–1759, 2018.

[3] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, Yunfei Bai, Matthew Kelcey, Mrinal Kalakrishnan, Laura Downs, Julian Ibarz, Peter Pastor, Kurt Konolige, Sergey Levine, and Vincent Vanhoucke. Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping. In *International Conference on Robotics and Automation (ICRA)*, pages 4243–4250, 2018.

[4] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4754–4765, 2018.

[5] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 29, pages 64–72. Curran Associates, Inc., 2016.

[6] Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3601–3610, 2017.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative

Adversarial Nets. In *Advances in Neural Information Processing Systems (NIPS)*, volume 27, pages 2672–2680, 2014.

[8] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. In *International Conference on Machine Learning (ICML)*, pages 2555–2565, 2019.

[9] Nicolas Heess, Jonathan J Hunt, Timothy P Lillicrap, and David Silver. Preparing for the Unknown: Learning a Universal Policy with Online System Identification. In *Robotics: Science and Systems (RSS)*, 2017.

[10] Stephen James, Andrew J. Davison, and Edward Johns. Transferring End-to-End Visuomotor Control from Simulation to Real World for a Multi-Stage Task. In *Conference on Robot Learning (CoRL)*, pages 334–343, 2017.

[11] Stephen James and Edward Johns. 3D Simulation for Robot Arm Control with Deep Q-Learning. In *NIPS 2016 Workshop (Deep Learning for Action and Interaction)*, 2016.

[12] Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12627–12637, 2019.

[13] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[14] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. In *International Conference on Learning Representations (ICLR)*, 2016.

[15] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J. Pal, and Liam Paull. Active Domain Randomization. In *Conference on Robot Learning (CoRL)*, pages 1162–1176, 2019.

[16] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[17] Anusha Nagabandi, Kurt Konoglie, Sergey Levine, and Vikash Kumar. Deep Dynamics Models for Learning Dexterous Manipulation. In *Conference on Robot Learning (CoRL)*, pages 1101–1112, 2019.

[18] Alexander Pashevich, Robin Strudel, Igor Kalevatykh, Ivan Laptev, and Cordelia Schmid. Learning to Augment Synthetic Images for Sim2Real Policy Transfer. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 2651–2657, 2019.

[19] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *International Conference on Robotics and Automation (ICRA)*, pages 3803–3810, 2018.

[20] Fereshteh Sadeghi and Sergey Levine. CAD2RL: Real Single-Image Flight Without a Single Real Image. In *Robotics: Science and System (RSS)*, 2017.

[21] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, 2017.

[22] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 5026–5033, 2012.

[23] Rishi Veerapaneni, John D. Co-Reyes, Michael Chang, Michael Janner, Chelsea Finn, Jiajun Wu, Joshua Tenenbaum, and Sergey Levine. Entity Abstraction in Visual Model-Based Reinforcement Learning. In *Conference on Robot Learning (CoRL)*, pages 1439–1456, 2019.

[24] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. In *IEEE Transactions on Image Processing*, volume 13, pages 600–612, 2004.

[25] C. Karen Liu Greg Turk Wenhao Yu, Jie Tan. Memory-based control with recurrent neural networks. In *NIPS 2015 Workshop (Deep Reinforcement Learning)*, 2015.

[26] Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Information-theoretic model predictive control: Theory and applications to autonomous driving. In *IEEE Transactions on Robotics*, volume 34, pages 1603–1622, 2018.

[27] Yufei Ye, Dhiraj Gandhi, Abhinav Gupta, and Shubham Tulsiani. Object-centric forward modeling for model predictive control. In *Conference on Robot Learning (CoRL)*, pages 100–109, 2019.

付録

A. 学習モデルを用いたプランニングと制御

A.1 モデル予測制御

システムの現在の状態を $\mathbf{s}_t$，制御入力を $\mathbf{u}_t$ としたとき，モデル学習では，環境の状態遷移 $\mathbf{s}_{t+1} = f(\mathbf{s}_t, \mathbf{u}_t)$ から得られた N 個の観測データ $\mathcal{D} = \{(\mathbf{s}_t, \mathbf{u}_t), \mathbf{s}_{t+1}\}_{n=1}^{N}$ を用いて，真の状態遷移モデル f を近似するような $\tilde{f}$ を決定的，もしくは確率的に学習する．そして，状態遷移モデルの学習が完了すれば，学習した $\tilde{f}$ によって環境の状態予測が可能となり，$\tilde{f}$ が確率的なモデルである場合，

$$\mathbf{u}_{t:t+H-1} = \underset{\mathbf{u}_{t:t+H-1}}{\arg\max} \sum_{\tau=t}^{t+H-1} \left\langle r(\mathbf{s}_\tau, \mathbf{u}_\tau) \right\rangle_{\tilde{f}} \tag{54}$$

として予測状態に基づく期待報酬の計算と，期待報酬に基づく最適制御入力系列の決定が可能となる．ここで，H はプラニングホライゾン，r は報酬である．このように，環境のモデルを用いて状態予測を行い最適な制御入力を決定する方法をモデル予測制御（Model Predictive Control: MPC）と呼ぶ.

A.2 Cross-entropy Method

MPC には様々な種類の方法があるが，ロボットの行動学習の研究においては Cross-entropy Method（CEM）が広く使われている [23, 4, 27, 26]．CEM では，ガウス分布から N 個の制御入力系列をサンプリングし，その中で報酬が最も高い上位 J 個の制御入力系列（エリート系列）を用いてガウス分布の平均と分散を更新する．そして，エリート系列によるガウス分布の更新を M 回繰り返した後，最終的に得られたガウス分布の平均が最適な制御入力系列として得られる．CEM の更新手順は式 (55) から式 (58) のように表すことができる．

$$\mathbf{U}_i = \{\mathbf{u}_0^i \dots \mathbf{u}_{H-1}^i\}, \text{ where } \ \mathbf{u}_t^i \sim \mathcal{N}(\boldsymbol{\mu}_t^m, \boldsymbol{\Sigma}_t^m) \ \ \forall i \in N, t \in 0 \dots H-1 \tag{55}$$

$$\mathbf{U}_{\text{elites}} = \text{sort}(\mathbf{U}_i)[-J:] \tag{56}$$

$$\boldsymbol{\mu}_t^{m+1} = \alpha * \text{mean}(\mathbf{U}_{\text{elites}}) + (1-\alpha)\boldsymbol{\mu}_t^m \ \ \forall t \in 0 \dots H-1 \tag{57}$$

$$\boldsymbol{\Sigma}_t^{m+1} = \beta * \text{var}(\mathbf{U}_{\text{elites}}) + (1-\beta)\boldsymbol{\Sigma}_t^m \ \ \forall t \in 0 \dots H-1 \tag{58}$$

ここで，$\mathbf{U}_i$ は H 個の $\mathbf{u}_t$ から成る 1 つの制御入力系列，$\mathbf{U}_{\text{elites}}$ はエリート系列である．また，$\boldsymbol{\mu}_t^m$ と $\boldsymbol{\Sigma}_t^m$ は m 回目の更新時のプランニングホライゾンの t ステップ目に対応するガウス分布の平均と分散であり，α, β はガウス分布の平均と分散の更新度合いを決定するハイパーパラメータである．

図 17 に 2 次元空間での CEM による最適化の例を示す

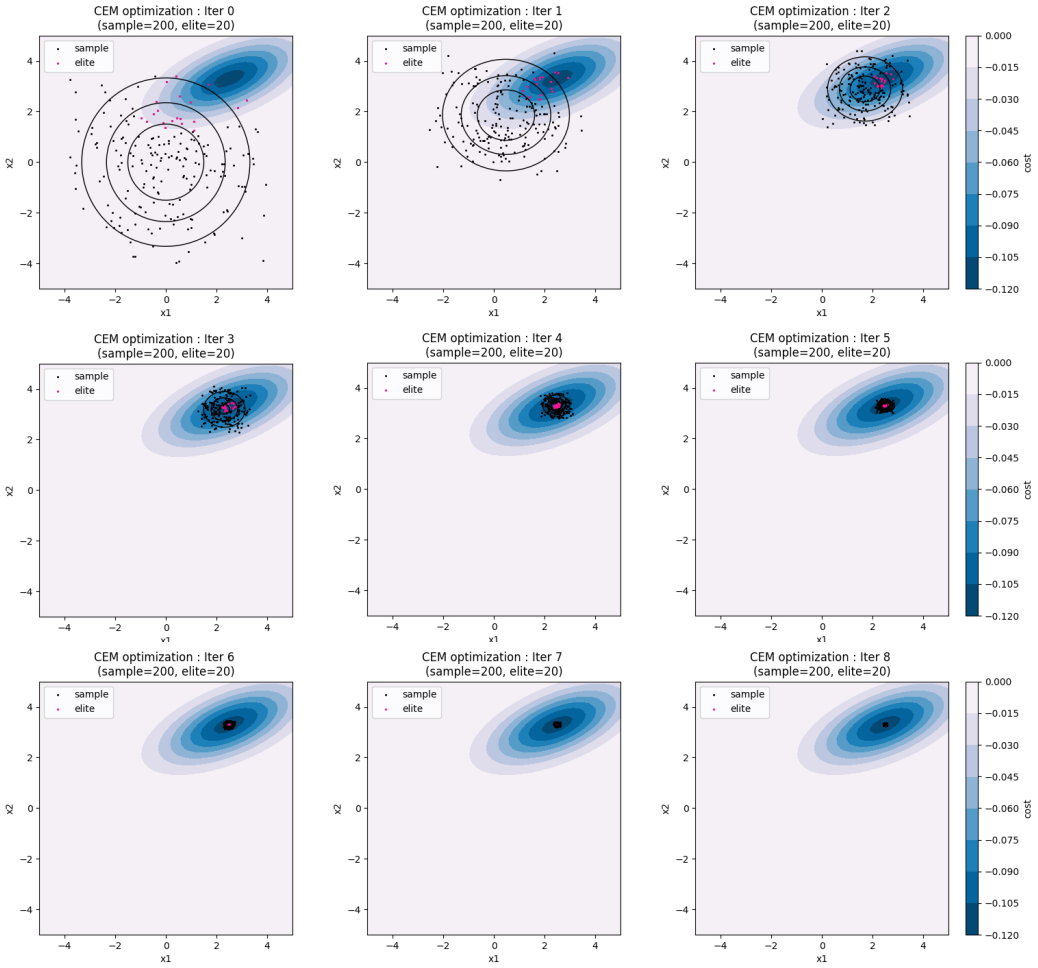


Fig. 17: Example of CEM Optimization

B. カルマン変分オートエンコーダ

B.1 モデル

カルマン変分オートエンコーダ（Kalman Variational Auto-Encoder: KVAE）[6] は LGSSM と VAE を組み合わせたモデルであり，高次元データである画像系列から動的モデルを学習するフレームワークである．KVAE では，画像系列中の物体の動きや相互作用を説明する重要な情報は，もとの画像の次元よりも小さい次元の多様体上にあるという仮説に基づき，画像の潜在表現と動的モデルの潜在表現を切り分けることを考える．ここで画像の潜在表現とは，例えば画像中の物体の位置情報のようなものである．ボールが動く動画から動的モデルを学習する場合，各フレームでの画像からボールの座標を抽出することができれば，抽出された座標を用いて動的モデルの学習を行うことができると考えられる．

観測画像系列を $\mathbf{x} = [\mathbf{x}_1, .., \mathbf{x}_T]$ としたとき，KVAE では画像の潜在表現と動的モデルの潜在情報を切り分けるため，各時刻における画像 $\mathbf{x}_t$ は VAE のエンコーダによって低次元空間 $\mathbf{a}_t$ にエンコードされ，エンコードされた $\mathbf{a}_t$ は動的モデル（LGSSM）にとっての疑似観測として使用される．このような画像の潜在表現と動的モデルの潜在表現が切り分けられた構造は「Disentangled な構造」と呼ばれる．KVAE のグラフィカルモデルは図 18 のように表される．

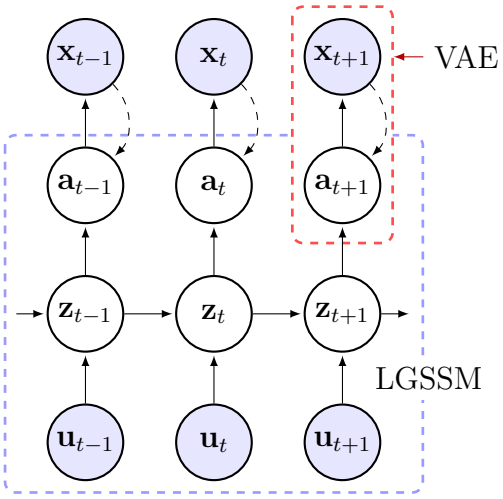


Fig. 18: Graphical Model of KVAE

潜在表現 $\mathbf{a} = [\mathbf{a}_1, .., \mathbf{a}_T]$ から観測画像系列への生成モデルは

$$p_\theta(\mathbf{x}|\mathbf{a}) = \prod_{t=1}^{T} p_\theta(\mathbf{x}_t|\mathbf{a}_t) \tag{59}$$

として分解し，$p_\theta(\mathbf{x}_t|\mathbf{a}_t)$ はパラメータ θ を持つ DNN でモデル化する．また同時に，$\mathbf{a}$ は LGSSM によってモデル化されるため，事前分布は式 (5) より

$$p_\gamma(\mathbf{a}|\mathbf{u}) = \int p_\gamma(\mathbf{a}|\mathbf{z})\, p_\gamma(\mathbf{z}|\mathbf{u})\, \mathrm{d}\mathbf{z} \tag{60}$$

と表され，同時確率分布は式 (61) のように表される．

$$p(\mathbf{x}, \mathbf{a}, \mathbf{z}|\mathbf{u}) = p_\theta(\mathbf{x}|\mathbf{a})\, p_\gamma(\mathbf{a}|\mathbf{z})\, p_\gamma(\mathbf{z}|\mathbf{u}) \tag{61}$$

従って，モデルの推論において高次元の画像を直接介すことなく，$\mathbf{z}$ と $\mathbf{a}$ の潜在空間のみで長期の予測が可能である．また，$\mathbf{z}$ と $\mathbf{a}$ を切り分けていることで，LGSSM の推論における計算量が大きく削減されている．LGSSM の推論では式 (4) における観測行列 $\mathbf{C}_t$ の逆行列計算が必要となるため，$\mathbf{C}_t$ の次元を D としたとき $\mathcal{O}(D^3)$ の計算を要する．しかし，$\mathbf{C}_t$ の次元は観測データの次元に依存するため，仮に画像のような高次元データを直接観測データとしてモデル化すれば，逆行列をそもそも計算できないという問題が生じる．これに対し，$\mathbf{a}$ という画像の潜在状態を仮定し，LGSSM にとっての観測としてモデル化することで，画像の次元に依らない逆行列計算が可能となる．また，一般的に LGSSM ではプロセスノイズと観測ノイズを完全な共分散行列でモデル化するが，KVAE では $\mathbf{a}$ は VAE の生成モデルの事前分布としての役割も果たすため，等方的な共分散行列を考える．LGSSM で非線形な動的モデルを学習するためのトリックは B.3 節で説明する．

B.2 学習

潜在変数の事後分布 $p(\mathbf{a}, \mathbf{z}|\mathbf{x}, \mathbf{u})$ は $p_\theta(\mathbf{x}_t|\mathbf{a}_t)$ が DNN でモデル化されるため解析的に計算することができない．従って，学習は対数周辺尤度

$$\mathcal{L} = \sum_{n}^{N} \log p_{\theta\gamma}(\mathbf{x}_{(n)}|\mathbf{u}_{(n)}) \tag{62}$$

の変分下限を最大化することで行う．ここで，N は系列数を表す．説明の簡略化のため，以後の説明では系列インデックス n は省略して表記する．変分分布を

$q(\mathbf{a}, \mathbf{z}|\mathbf{x}, \mathbf{u})$ とすると対数周辺尤度の変分下限は

$$\log p(\mathbf{x}|\mathbf{u}) = \log \int p(\mathbf{x}, \mathbf{a}, \mathbf{z}|\mathbf{u}) \mathrm{d}\mathbf{a}\mathrm{d}\mathbf{z} \tag{63}$$

$$\geq \left\langle \log \frac{p_\theta(\mathbf{x}|\mathbf{a}) p_\gamma(\mathbf{a}|\mathbf{z}) p_\gamma(\mathbf{z}|\mathbf{u})}{q(\mathbf{a}, \mathbf{z}|\mathbf{x}, \mathbf{u})} \right\rangle_{q(\mathbf{a},\mathbf{z}|\mathbf{x},\mathbf{u})} \tag{64}$$

$$= \mathcal{F}(\theta, \gamma, \phi) \tag{65}$$

として得られる．ここで，ϕ は変分分布が持つパラメータである．変分分布はカルマンスムーザの結果を活用して

$$q(\mathbf{a}, \mathbf{z}|\mathbf{x}, \mathbf{u}) = q_\phi(\mathbf{a}|\mathbf{x}) p_\gamma(\mathbf{z}|\mathbf{a}, \mathbf{u}) = \textstyle\prod_{t=1}^{T} q_\phi(\mathbf{a}_t|\mathbf{x}_t)\, p_\gamma(\mathbf{z}|\mathbf{a}, \mathbf{u}) \tag{66}$$

とおく．このように変分分布をおくことで，$\mathbf{x}_t$ をエンコードすればその後はカルマンスムーザが実行できるため，LGSSM の利点を受け継ぐことができる．また，$q_\phi(\mathbf{a}|\mathbf{x})$ は DNN でモデル化される．これより，変分下限は改めて，

$$\mathcal{F}(\theta, \gamma, \phi) = \left\langle \log \frac{p_\theta(\mathbf{x}|\mathbf{a})}{q_\phi(\mathbf{a}|\mathbf{x})} + \left\langle \log \frac{p_\gamma(\mathbf{a}|\mathbf{z}) p_\gamma(\mathbf{z}|\mathbf{u})}{p_\gamma(\mathbf{z}|\mathbf{a}, \mathbf{u})} \right\rangle_{p_\gamma(\mathbf{z}|\mathbf{a},\mathbf{u})} \right\rangle_{q_\phi(\mathbf{a}|\mathbf{x})} \tag{67}$$

と書くことができる．この変分下限はサンプル $\{\widetilde{\mathbf{a}}_i, \widetilde{\mathbf{z}}_i\}_{i=1}^{I}$ を用いたモンテカルロ積分により，式 (68) で推定することができる．

$$\hat{\mathcal{F}}(\theta, \gamma, \phi) = \frac{1}{I} \sum_i \{\log p_\theta(\mathbf{x}|\widetilde{\mathbf{a}}_i) + \log p_\gamma(\widetilde{\mathbf{a}}_i, \widetilde{\mathbf{z}}_i|\mathbf{u}) - \log q_\phi(\widetilde{\mathbf{a}}_i|\mathbf{x}) - \log p_\gamma(\widetilde{\mathbf{z}}_i|\widetilde{\mathbf{a}}_i, \mathbf{u})\} \tag{68}$$

B.3 動的パラメータネットワーク

LGSSM の構造は変分分布が $p_\gamma(\mathbf{z}|\mathbf{a}, \mathbf{u})$ を計算するためには非常に都合がいいもの，このままでは非線形な動的モデルを学習できない．これに対し KVAE では，各時刻でそれまでの疑似観測 $\mathbf{a}_{0:t-1}$ に依存して変化するパラメータ γ_t を出力する動的パラメータネットワークを導入することで，非線形な動的モデルの学習に対応する．カルマンスムーザによる $p_\gamma(\mathbf{z}_t|\mathbf{a}, \mathbf{u})$ の厳密な事後分布の計算においては $\gamma_t = [\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t]$ は定数である必要はないため，動的パラメータネットワークから出力される時変な γ_t を用いることは計算上問題ない．

一般的に，時刻 t での動的モデルの変化はシステムの過去の遷移に依存すると考えられる．例えば，ボールが壁に衝突して跳ね返る動画があった場合，時刻 $t-1$ までの画像系列が与えられ，時刻 $t-1$ においてボールが壁に衝突している状態であれば，その先の時刻 t ではボールが跳ね返って進む方向が変化するという遷移が予測できる．従って，γ_t は時刻 $t-1$ から時刻 t への動的モデルの変化を説明するようなパラメータとなっているべきである．上記の考えに基づき，γ_t を $\mathbf{a}_{0:t-1}$ の関数として構成することで，潜在変数の事前分布は

$$p_\gamma(\mathbf{a}, \mathbf{z}|\mathbf{u}) = \prod_{t=1}^{T} p_{\gamma_t(\mathbf{a}_{0:t-1})}(\mathbf{a}_t|\mathbf{z}_t) \cdot p(\mathbf{z}_1) \prod_{t=2}^{T} p_{\gamma_t(\mathbf{a}_{0:t-1})}(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{u}_t) \tag{69}$$

と書くことができる．ここで，$\mathbf{a}_0$ はデータから学習するパラメータである．

動的パラメータネットワークは各時刻においてエンコードされた $\mathbf{a}_t$ を入力として受け取る LSTM（Long Short-Term Memory）と，LSTM の出力を受け取り softmax 変換を行う全結合層から成り，

$$\boldsymbol{\alpha}_t = \mathrm{softmax}(\mathbf{d}_t) \tag{70}$$

$$\mathbf{d}_t = \mathrm{LSTM}(\mathbf{a}_{t-1}, \mathbf{d}_{t-1}) \tag{71}$$

という構造を取る．初期時刻における $\mathbf{d}_0$ のみ全ての要素が 0 のベクトルで初期化される．動的パラメータネットワークの出力 $\boldsymbol{\alpha}_t = [\alpha_t^{(1)}, .., \alpha_t^{(K)}]$ は $\sum_{k=1}^{K} \alpha_t^{(k)}(\mathbf{a}_{0:t-1}) = 1$ という制約を満たす K 次元のベクトルであり，K 個の異なる基底を重みづけて足し合わせるために使われる．従って，$\gamma_t = [\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t]$ の各パラメータは

$$\mathbf{A}_t = \sum_{k=1}^{K} \alpha_t^{(k)}(\mathbf{a}_{0:t-1})\mathbf{A}^{(k)} \tag{72}$$

$$\mathbf{B}_t = \sum_{k=1}^{K} \alpha_t^{(k)}(\mathbf{a}_{0:t-1})\mathbf{B}^{(k)} \tag{73}$$

$$\mathbf{C}_t = \sum_{k=1}^{K} \alpha_t^{(k)}(\mathbf{a}_{0:t-1})\mathbf{C}^{(k)} \tag{74}$$

として表される．K 個の基底となる $\mathbf{A}^{(k)}, \mathbf{B}^{(k)}, \mathbf{C}^{(k)}$ はデータセット全体に渡って共通したものを学習する．これは，時不変な行列が時変な重み $\boldsymbol{\alpha}_t$ によって組み合わされた K 個の異なる LGSSM の混合として解釈できる．また，実際に各 K の組み合わせ $\{\mathbf{A}^{(k)}, \mathbf{B}^{(k)}, \mathbf{C}^{(k)}\}$ は異なる動的モデルをモデル化する．

動的パラメータネットワークの構造を図示したものを図 19 に示す.

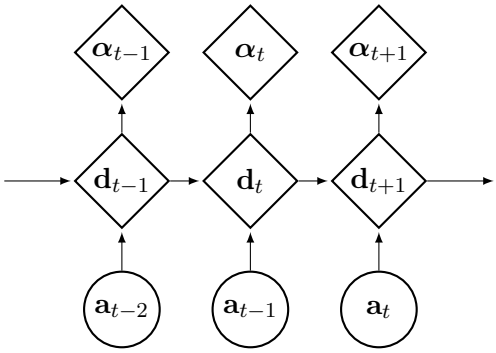


Fig. 19: Structure of Dynamics Parameter Network in KVAE

C. バルブ押し込みタスクにおける性能評価（シミュレーション 4）の追加情報

C.1 CEM の設定

Table 3: Settings of CEM in Simulation 4 at Section5.6

Hyperparameter	**Vlaue**
number of samples	3000
number of elite	10
horizon	5
number of update	10
initial mean per dim	0
initial variance per dim	1
α	0.3
β	0.1
clipping parameter for lower bound	-1.5
clipping parameter for upper bound	1.5

C.2 CEM による制御入力系列の最適化

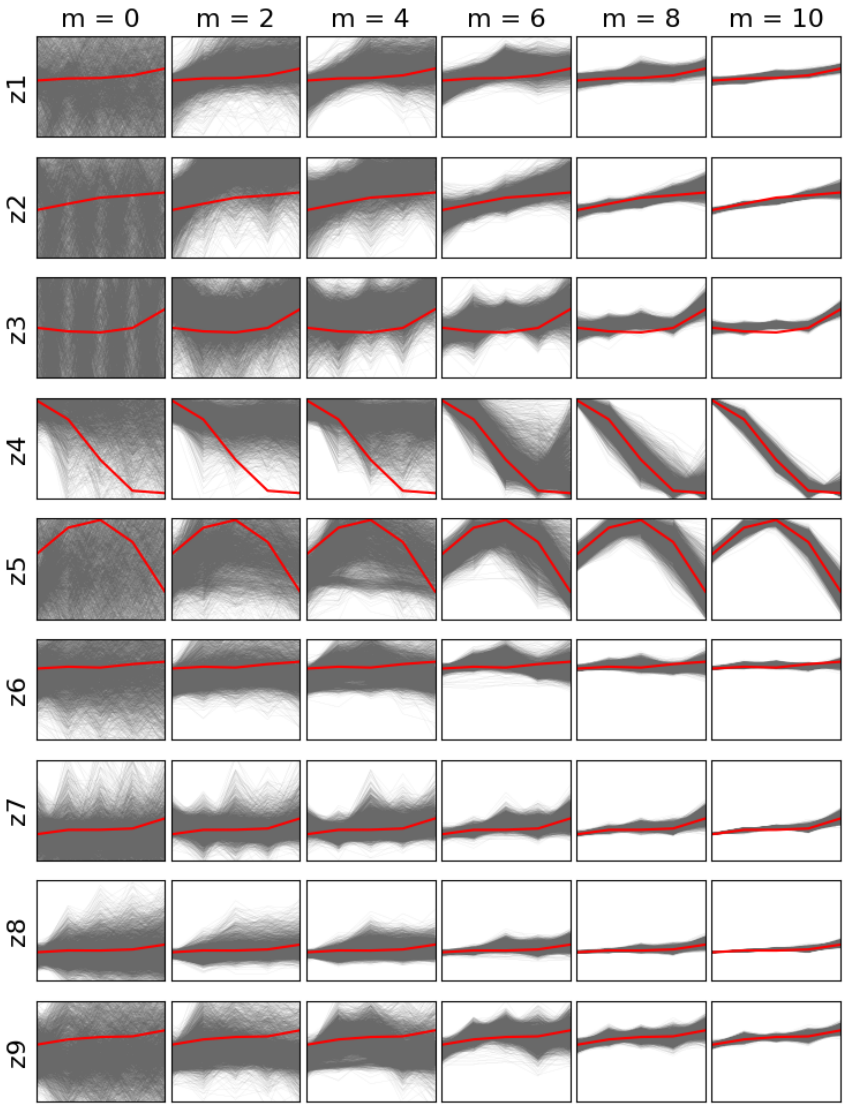


Fig. 20: Example of Optimization of Control Sequences by CEM

D. V-RCANのネットワーク構造

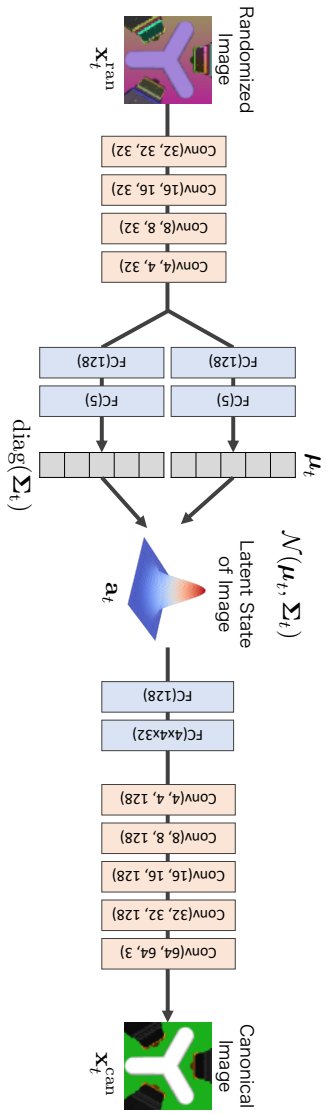


Fig. 21: V-RCAN structure