

NAIST-IS-MT1651088

修士論文

見かけ上の誤差に頑健な方策転移学習

則永 悠

2018年3月15日

奈良先端科学技術大学院大学
情報科学研究科 情報科学専攻

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

則永 悠

審査委員：

杉本 謙二 教授	(主指導教員)
小笠原 司 教授	(副指導教員)
松原 崇充 准教授	(副指導教員)
小蔵 正輝 助教	(副指導教員)
小林 泰介 助教	(副指導教員)

見かけ上の誤差に頑健な方策転移学習*

則永 悠

内容梗概

近年、視覚情報と行動の規則である方策の対応を学ぶ強化学習が注目を集めている。実環境での学習は高コストなため、シミュレーションで学習した方策を実環境に転移したい。しかし、シミュレーションと実環境間の見かけ上の誤差により、end-to-end な方策を実環境に転移することは難しい。本研究では、シミュレーションで学習をした end-to-end な方策を、実環境に転移する手法の提案を目標とする。シミュレーションと実環境に共通する特徴量で制御するよう学習することで、実環境画像を入力しても制御性能が劣化せず、シミュレーションから実環境へ方策を転移することが出来ると考えられる。本研究では、シミュレーションと実環境の見かけ上の誤差と方策の性能低下に頑健な特徴量を抽出し、抽出された特徴量でシミュレーションで学習を行うことで、実環境に転移可能な方策を学習する手法を提案した。特徴量抽出の付加的な基準として、データの分散の保存と報酬値と特徴量の相関の保存の2種類を提案した。空間的ピラミッドマッチングにより汎用特徴量を抽出し、転移成分分析により、両環境に共通する特徴量を抽出した。抽出した特徴量を用いて、シミュレーションで強化学習を行い、実環境に転移可能な方策を生成した。シミュレーションで双腕ロボットによるリーチングタスクを学習し、実環境に転移する実験で提案手法の有効性の検証を行い、データの分散を保存するよう特徴抽出する提案法の有効性を確認した。

キーワード

強化学習, 方策転移学習, 転移成分分析, 空間的ピラミッドマッチング, EM 方策探索

*奈良先端科学技術大学院大学 情報科学研究科 修士論文, 情報科学専攻 NAIST-IS-MT1651088, 2018年3月15日.

Policy transfer learning with appearance errors^{*}

Norinaga Yu

Abstract

Recently, end-to-end reinforcement learning that learns relationship between image and policy has been drawing much attention. Reinforcement learning in real world needs a huge cost, so it is desirable to transfer the policy learned from simulation to the real world. It is difficult to transfer the policy because of appearance errors between simulation and real world. In this research, we aim to propose a method of transferring end-to-end policy learned in simulation without learning in real world. Learning policies to control by features commonly existing between simulation and real world makes it possible to transfer the policy to real world without performance degradation. We proposed a framework to transfer a policy to real world with the features that is robust to these appearance errors. We proposed two feature extraction criteria, preservation of data variance and correlation between rewards and features. We extract general purpose features from images using spatial pyramid matching and extracting common features in both environments by using transfer component analysis. We generated a policy which can be transferred to real world by learning a policy in simulation with extracted features. In the experiment, we confirmed the effectiveness of the proposed method of feature extraction to preserve the variance of data.

Keywords:

Reinforcement Learning, Policy Transfer Learning, Transfer Component Analysis, Spatial Pyramid Matching, EM Algorithm

^{*}Master's Thesis, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-MT1651088, March 15, 2018.

目次

1. 序論	1
1.1 研究背景	1
1.2 end-to-end な方策の転移に関する従来研究	2
1.2.1 詳細なモデル等を用いた従来研究	2
1.2.2 詳細なモデル等を用いない従来研究	5
1.3 研究目的	5
1.4 本論文の構成	7
2. 問題設定	8
2.1 見かけ上の誤差について	8
2.2 問題解決の着想	9
3. 準備	13
3.1 強化学習概要	13
3.2 EM 方策探索	13
3.3 空間的ピラミッドマッチング (Spatial Pyramid Matching:SPM) .	15
3.4 転移成分分析	16
3.4.1 Maximum Mean Discrepancy	17
3.4.2 Hilbert-Schmidt Independence Criterion	18
3.4.3 転移成分分析	18
4. 提案法：方策転移学習	21
4.1 分散を基準とした特徴量による方策転移学習概要	21
4.2 報酬値を基準とした特徴量による方策転移学習概要	22
5. 実験	23
5.1 見かけ上の誤差を含むシミュレーション環境に方策転移	23
5.1.1 実験システム	23
5.1.2 実験設定	26

5.1.3	実験結果	30
5.2	実環境に方策転移	36
5.2.1	実験システム	36
5.2.2	実験設定	37
5.2.3	実験結果	40
6.	結論	46
6.1	まとめ	46
6.2	今後の課題	46
	謝辞	49
	参考文献	50

目 次

1	実環境と非常によく似た 3D モデルを用いた方策転移	4
2	3D モデルと深度画像を利用して方策転移	4
3	AR マーカーを利用した方策転移	4
4	マルチタスク学習を利用した方策転移. シミュレーションで学習したネットワークの各層の出力を, 実環境のネットワークに入力して学習した知識を転移している	6
5	ランダムな変化を加えたシミュレーション環境と実環境	6
6	シミュレーションで学習した方策を実環境に直接転移	8
7	シミュレーションと実環境の見かけ上の誤差	9
8	両環境の観測結果から共通する特徴量を抽出	11
9	分散を考慮した特徴抽出の例	11
10	分解レベル 2 の SPM による特徴抽出の一例. 例では画像は丸, 三角, 四角の三つの特徴に離散化されている. 各レベルにの各グリッドにおいてそれぞれの特徴量を数え上げ, ヒストグラムとして表現する	16
11	分散を基準とした特徴量による方策転移学習概要	21
12	報酬値を基準とした特徴量による方策転移学習概要	22
13	方策学習環境全体図	23
14	方策転移先の環境全体図	24
15	シミュレーション環境俯瞰図	24
16	二種類の経由地点	25
17	手順 2 における, $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{error}}$ の対応の例 (上)	28
18	手順 2 における, $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{error}}$ の対応の例 (下)	29
19	実験結果 (シミュレーション)	30
20	タスク実行例 (上)	31
21	タスク実行例 (下)	32
22	類似度計測用画像 (元環境)	34
23	類似度計測用画像 (転移先)	34

24	各手法による特徴量の類似度の比較	34
25	Baxter	36
26	kinect	37
27	俯瞰図	37
28	$\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{real}}$ の対応の例 (上)	38
29	$\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{real}}$ の対応の例 (下)	39
30	実験結果 (実環境)	40
31	タスク実行例 (上)	41
32	タスク実行例 (下)	42
33	類似度計測用画像 (シミュレーション)	43
34	類似度計測用画像 (実環境)	43
35	各手法による特徴量の類似度の比較	44

表 目 次

1. 序論

1.1 研究背景

近年、実ロボットによる強化学習の需要が高まっている。従来のロボットの制御では、どのような状況でどのような行動を取るべきかという行動のルールを人間が設計し、そのルールに従い制御を行ってきた。そのため、ありとあらゆる状況を想定して行動のルールを設計する必要があり、設計者の負担が非常に大きかった。また、想定されていない環境に対応する臨機応変な行動を取ることができないという問題もあった。加えて、多くの実環境においては不確実性や計測不能な未知パラメータが存在するため、タスクの達成方法が完全に自明ではなかった。従来のロボットの制御と比べると、強化学習では、達成してほしいタスクを報酬として指示しておくだけで、ロボットが試行錯誤の中から行動のルールである方策を自動で学習するため、設計者の負担が比較的小さい。また、人が設計したルールより優れた方策を学習する可能性がある、不確実性を含む環境に対応できるといったメリットがある。

その強化学習の中でも特に、視覚情報と行動の対応を学ぶ end-to-end な強化学習が注目を集めている。[1, 2] 通常の強化学習では、まず画像からオブジェクトの位置や姿勢等の検出を行うなど、各種センサから得られるデータに前処理を行い、入力とする必要がある。そのため、制御周期が短いと前処理が間に合わない可能性がある。また、各種センサからどのような情報を前処理により抜き出し入力とするかは人手で決めるため、入力がタスクを実行するのに十分でない場合は、学習により得られる方策がヒューリスティックになる可能性がある。end-to-end な強化学習では、センサ情報から行動への写像（方策）を直接学習するため、前処理が必要ない。センサ情報としては特に画像が用いられる場合が多い。また、制御に必要な特徴が学習の中で入力から自動的に選ばれるので、タスクの実行に必要な情報がすべて画像に含まれる場合は、学習により得られる方策はヒューリスティックにならない。

しかし、実ロボットによる end-to-end な強化学習には大きな問題がある。それは、実ロボットの強化学習は時間や費用等のコストがかかりすぎる点である。

Levine らはロボットハンドに約 10 種類の物を掴むタスクを学習させたが、学習が終わるまでに 11 台のロボットハンドを約 4 週間動作させ続ける必要があった [3]. また, Pinto らは, 最適なオブジェクトの把持方法をロボットに学習させたが, 学習におよそ 700 時間をかけて 5 万回の試行を必要としたことを報告している [4]. 物を掴むというシンプルなタスクを学習するのにこれだけのコストがかかるので, 複雑なタスクを学習するには更なるコストがかかるものと考えられる. この問題を避けるために, 実ロボットでの学習を行わずシミュレーションで方策を学習し, 学習した方策を実環境でそのまま利用する, 直接転移という方法も考えられる. しかし, end-to-end な強化学習で得られた方策を直接転移することは難しい. なぜなら, シミュレーションと実環境には見かけ上の誤差があるので, その誤差が原因となり, 実環境で方策を利用した際に, 制御性能が大幅に劣化しタスクを実行できない恐れがあるためである. そのため, シミュレーションで学習された end-to-end な方策を実ロボットで利用可能にするための機械学習の枠組みが期待されている.

1.2 end-to-end な方策の転移に関する従来研究

シミュレーションで学習した方策を実環境に転移する研究を大別すると, 実ロボットや実オブジェクトによく似たシミュレーションモデルやマーカーを用いることで, なるべく見かけ上の誤差を少なくしたり, 見かけ上の誤差の影響を受けない設定で学習するものと, よく似たシミュレーションモデルやマーカーを用いず, 見かけ上の誤差が大きい環境で学習するものがある. 特に, よく似たモデルを用い, 見かけ上の誤差をなるべく小さくして学習するものが多い.

1.2.1 詳細なモデル等を用いた従来研究

James らは, 図 1 のような実環境に非常によく似たモデルをシミュレーションで作成し, そのモデルを用いてシミュレーション上の画像を入力としてピッキング動作を学習し, 実環境への方策転移を行った [5]. 図 1 のシミュレーション環境で 2000 回以上の試行により, 赤のオブジェクトを持ち上げるタスクを学習し,

実環境に方策転移を行った。実環境ではロボットはオブジェクトに手先を近づけることはできたものの、オブジェクトを持ち上げるには至らなかった。図1のような非常によく似たシミュレーション環境を用いてすら、見かけ上の誤差の影響により、タスクを十分には達成できないため、方策転移を行う際には、他のアプローチが必要であることが分かる。

また、Johns らは、ModelNet[6] と呼ばれる大規模なオブジェクトの3Dモデルセットを用いて、オブジェクトの深度画像中のピッキング位置とロボットハンドの姿勢の対応を学習した[7]。本研究で用いた深度画像は、色情報を失っているものの、その分シミュレーションと実環境での差異が小さいため、見かけ上の誤差を小さくして学習を行うことが出来る。本研究では実環境での学習を行わずに図2(b) 実環境からの深度画像から赤で示されるように把持位置を決定し、(c) のようにオブジェクトのピッキングに成功している。

加えて、Clavera らは、シミュレーション上で、見かけ上の誤差の影響のないオブジェクトの位置情報と、モデル化誤差の影響のないロボットハンドの関節角度との対応を学習した。実環境では図3のようにオブジェクトにAR マーカーを貼り付け、そこから位置情報を取得することで、実環境での学習を行わず方策転移を行い、オブジェクトを目標の位置までプッシングすることに成功している[8]。他の研究とは異なり、画像と行動の関係を学習するという意味での end-to-end ではないものの、AR マーカー以外の特別な装置を用いずに、実環境でのタスクの実行に成功しているのは注目すべき点といえる。また、リーチングやピッキングではなく、プッシングを実行している点も、他の研究とは異なる点である。

以上のように、多くの研究では、詳細なモデルやAR マーカー等を用いて、見かけ上の誤差をできる限り小さくするか、あるいはオブジェクトの位置情報と関節角度の様な、見かけ上の誤差の存在しない関係について強化学習を行うことで、シミュレーションから実環境へと方策転移を行っている。しかしながら、モデルの事前作成の必要や、深度画像を撮影できる装置が必要であるなどの制限が生じてしまっている。そのため、これらの手法は利用できる環境やタスクが限定的である。

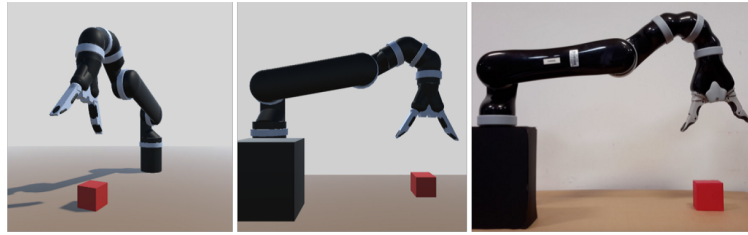


図 1 実環境と非常によく似た 3D モデルを用いた方策転移 [5]



(a) 実環境画像 (b) 深度画像と把持位置 (c) 把持に成功

図 2 3D モデルと深度画像を利用して方策転移 [7]

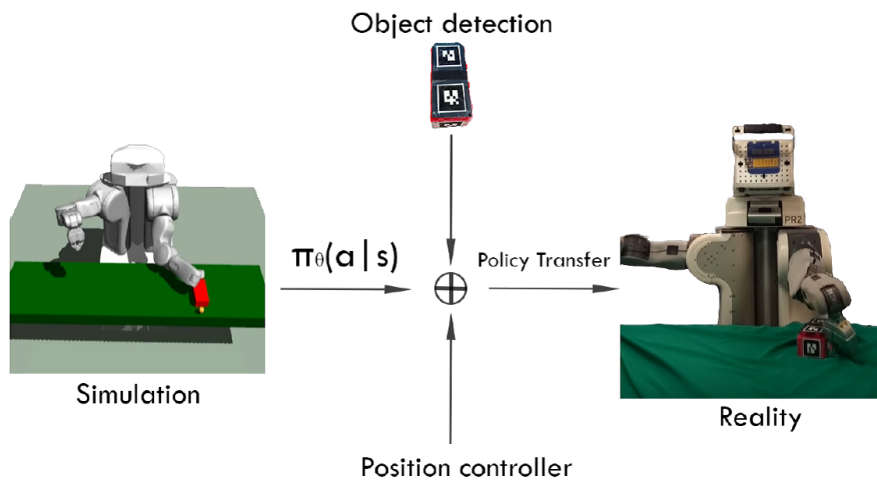


図 3 AR マーカーを利用した方策転移 [8]

1.2.2 詳細なモデル等を用いない従来研究

一方、実ロボットや実オブジェクトに似たシミュレーションモデルを用いない場合、見かけ上の誤差が大きくなるため、方策転移がより難しくなる。しかし、先に挙げた研究のように事前のモデル作成や使用する装置に制限がないため、様々な状況やタスクへの応用が考えられる。

Rusu らは、方策の転移を行うのではなく、シミュレーションと実環境での行動を別のタスクとみなし、シミュレーションで学習した知識の一部を実環境で利用することで、実環境での学習時間を短縮するというアプローチで見かけ上の誤差の問題解決に取り組んだ。本研究では、マルチタスク学習に利用される Progressive Neural Networks という図4のようなネットワークを用いてシミュレーションで学習した知識の一部を実環境に転移させ、実環境でのリーチングタスクの学習時間を短縮することに成功している [9]。実環境においても数時間の学習を要したが、実環境のみで学習を行った場合、タスクによっては数か月以上の時間がかかる点を考えると、大幅に学習時間を短縮出来たといえる。

Tobin らは、図5のようにシミュレーションにおいて色相や光源の位置や数、ノイズなどをランダムに変更し、様々な種類の見かけ上の誤差を含む画像を用いることで、様々な見かけ上の誤差に対応できるネットワークを学習した。結果として実環境画像での学習を必要とせず動作するオブジェクト検出器の学習に成功している [10]。本研究ではロボットの動作を学習する強化学習は行われていないが、複雑環境下で見かけ上の誤差を受けずに高精度にオブジェクト位置の検出に成功している点は、他の研究より優れている点である。

1.3 研究目的

従来研究では、シミュレーションにおいて、実オブジェクトや実ロボットによく似たモデルを用いたり、見かけ上の誤差が少ない深度画像を入力として、ディープニューラルネットワークで学習を行うものが多かった。このような学習法は、詳細なモデルや深度画像が撮影できるセンサが必要であるなどの制約が多い。また、ディープニューラルネットワークを用いることで、学習に大規模なデータが

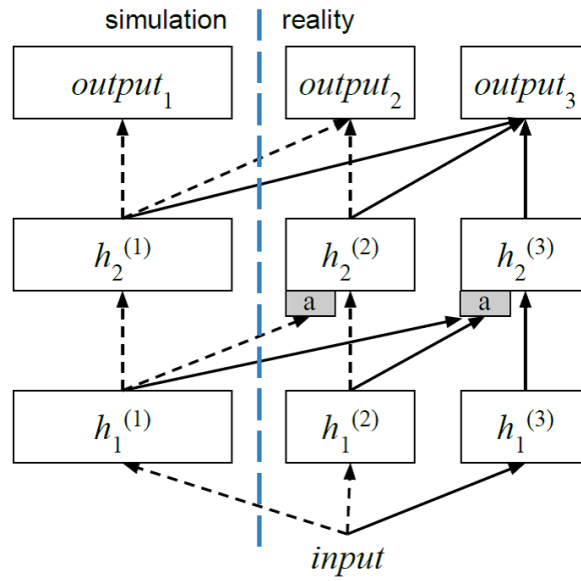


図 4 マルチタスク学習を利用した方策転移 [9]. シミュレーションで学習したネットワークの各層の出力を、実環境のネットワークに入力して学習した知識を転移している



図 5 ランダムな変化を加えたシミュレーション環境と実環境 [10]

必要になってしまう。データセットはシミュレーションにおいて作成されるので、実環境で学習するより時間は短縮できるものの、より少ないデータで学習できることが望ましい。また、タスクの成功の可否やその程度と特徴量の関係を陽に考慮していない場合、特徴量からタスクの実行に必要な情報が抜け落ちてしまう可能性がある。そのため、大規模データセットや詳細モデルを必要とせずに、シミュレーションと実環境に共通する特徴を、タスクの実行に必要な情報を保持したまま抽出できる事が望ましい。

本研究の目的は、シミュレーションで学習した方策を実環境に転移する機械学習の枠組みの提案である。本研究では、シミュレーションにおいて強化学習により獲得された方策を、見かけ上の誤差を伴う実環境に転移させる枠組みとして方策転移学習を提案する。提案手法では、シミュレーションと実環境の見かけ上の誤差と方策の性能低下に頑健な特徴量を抽出する。特徴量抽出の付加的な基準として、各データの分散の保存と、シミュレーションから得られる報酬値と特徴量の相関の保存の2種類を考える。特徴量の抽出手法として、転移成分分析 (Transfer Component Analysis: TCA) [11] と呼ばれる転移学習手法を利用する。抽出された特徴量を方策の入力として、シミュレーションにおいて再度強化学習を行うことで、実環境での学習を行わずに実環境に転移可能な方策を学習する。また、シミュレーションにおいて学習した方策を、見かけ上の誤差の伴うシミュレーション環境や実環境に転移することで、提案手法の有効性を検証する。

1.4 本論文の構成

本論文では、第2章にて、問題設定と問題解決のため着想について述べる。第3章では、2章で述べた着想を実現するための要素技術について説明する。第4章では、要素技術を用いて構成した提案法である方策転移学習について説明する。第5章では、提案法の有効性のために行った実験について説明する。

2. 問題設定

2.1 見かけ上の誤差について

はじめに、シミュレーションで学習した end-to-end な方策を実環境に直接転移する際の問題について述べる．図 6 のように，シミュレーション環境において，

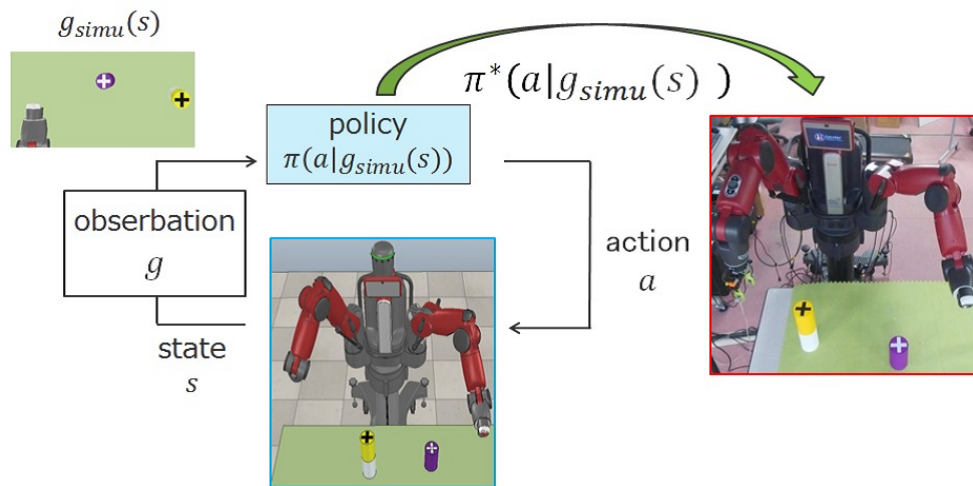


図 6 シミュレーションで学習した方策を実環境に直接転移

状態 s を観測し，観測結果 $g_{simu}(s)$ を生成，方策 $\pi(a|g_{simu}(s))$ へ入力することで行動 a を生成し，行動により状態 s が変化する，といった繰り返しの中で，シミュレーション環境においてタスクを十分に実行できる最適な方策 $\pi^*(a|g_{simu}(s))$ を学習できる．その方策を実環境に転移することで，実環境で学習を行わずにロボットにタスクを実行させる，シミュレーションから実環境への方策の直接転移ができると考えられる．しかしながらこの方法は成功しない．ここで，図 7 のようにシミュレーションと実環境に関して，ロボットの姿勢やオブジェクトの位置が同様である場合を考える．この時、シミュレーションと実環境において状態 s は同じであると解釈できる．一方で，両環境の状態 s を観測 g により，状態 s から各環境の観測結果 $g(s)$ へ写像すると，シミュレーションの観測結果 $g_{simu}(s)$ と，実環境の観測結果 $g_{real}(s)$ は一致しない．これは，シミュレーションが実環境を完全には再現できていないため，影や光沢，色合い等に違いが生じてしまうからであ

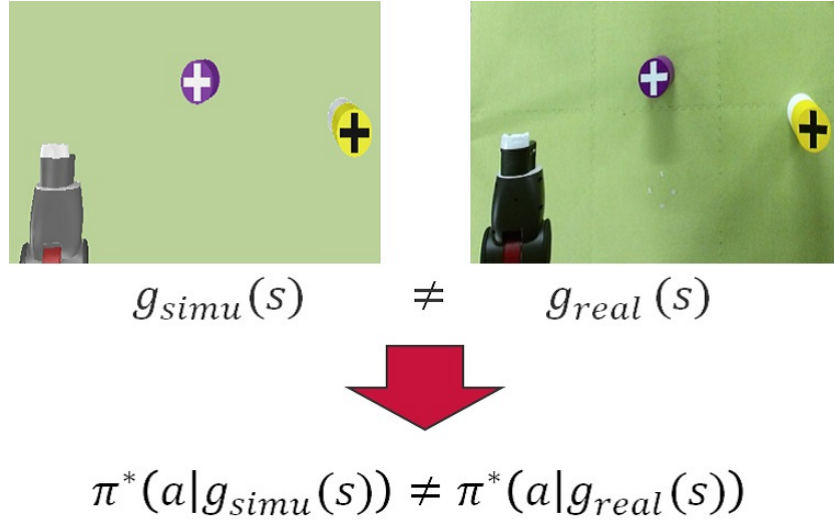


図 7 シミュレーションと実環境の見かけ上の誤差

る．この観測上の違いが両環境の見かけ上の誤差である．そのため，シミュレーションの観測結果 $g_{simu}(s)$ で学習して得られる最適な方策と，実環境の観測結果 $g_{real}(s)$ で学習して得られる最適な方策が一致しない．つまり， $g_{simu} \neq g_{real}$ であるため， $g_{simu}(s) \neq g_{real}(s)$ となり， $\pi^*(a|g_{simu}(s)) \neq \pi^*(a|g_{real}(s))$ となる．その結果，シミュレーションで学習して得られる方策 $\pi^*(a|g_{simu}(s))$ に，実環境の観測結果 $g_{real}(s)$ を入力しても，タスクを実行することが出来ない．

2.2 問題解決の着想

1 章でも述べたように，本研究では，シミュレーションで学習した方策を，見かけ上の誤差を伴う実環境に転移することを目標としている．シミュレーションと実環境では，色合いや光の反射などの見かけ上の違いがある．そのため，シミュレーションで学習した方策に実環境の観測結果，つまり実環境画像を入力した際に，制御性能が大幅に劣化するという問題がある．この問題に対し，シミュレーションと実環境の画像の共通して存在する特徴量に着目して制御するように方策を学習することで，実環境画像を入力した場合でも制御性能が劣化せず，シミュ

レーションから実環境へ方策を転移することが出来ると考えられる。

上記の着想を満たす特徴量は、以下の要件を満たす必要がある。

要件1 シミュレーションと実環境の画像から抽出される特徴量の差異が少ない

要件2 画像に含まれる重要な情報を失わないよう特徴抽出する

まず要件1に関して説明する。シミュレーションと実環境の画像に含まれる特徴量は、以下の3つがあるといえる。

特徴量1 シミュレーション画像のみに含まれる特徴量

特徴量2 実環境画像のみに含まれる特徴量

特徴量3 シミュレーションと実環境の画像の両方に含まれる特徴量

シミュレーション画像を用いて方策を学習するということは、特徴量1と3に基づいて方策を学習することである。そのため、この方策を実行するには特徴量1と3が必要となる。この方策に実環境画像を入力した場合、実環境画像には特徴量3が含まれているので、タスクを実行するのに必要な特徴量を一部含んでいるものの、特徴量1を含んでいないため、タスクを十分に達成できない可能性がある。また、シミュレーション画像を入力として方策を学習すると、特徴量1より特徴量3を重視して学習した場合、その方策に実環境画像を入力しても、学習したタスクをある程度を達成できるようにも思える。しかしながら、実環境画像を入力すると、実環境画像に含まれる特徴量2がノイズとして作用するため、制御性能が大幅に劣化し、タスクを十分に達成することが出来ない。そのため、シミュレーションで方策を学習する際には、特徴量3のみに基づいて学習する必要がある、学習した方策に実環境画像を入力する際には、特徴量3を実環境画像から抽出して入力する必要がある。以上から、両環境の画像から、できる限り差異の少ない特徴量を抽出する必要がある。

特徴抽出の概要を図8に表す。シミュレーションや実環境で観測を行うことで得られる観測結果 $g_{simu}(s)$ と $g_{real}(s)$ は一致しない。そこで、両環境の観測結果に共

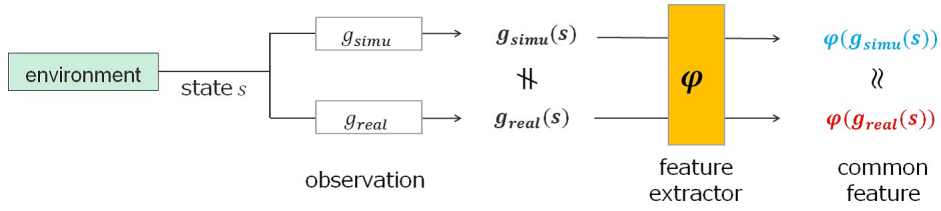


図 8 両環境の観測結果から共通する特徴量を抽出

通する特徴量を抽出する特徴抽出器 ϕ を用いることで、 $\phi(g_{simu}(s)) \approx \phi(g_{real}(s))$ となるように特徴抽出を行う。

次に、要件 2 に関して説明する。要件 1 の説明で共通する特徴量を抽出することは述べたが、画像から特徴量を抽出する際に、元のデータの散らばりの度合いを示す分散や、ロボットの位置や姿勢といったタスクの実行に必要な情報が、特徴抽出の際に失われる可能性がある。その場合、特徴量 3 であっても、学習が行えない、タスクを十分に実行することができなくなるなどの恐れがある。

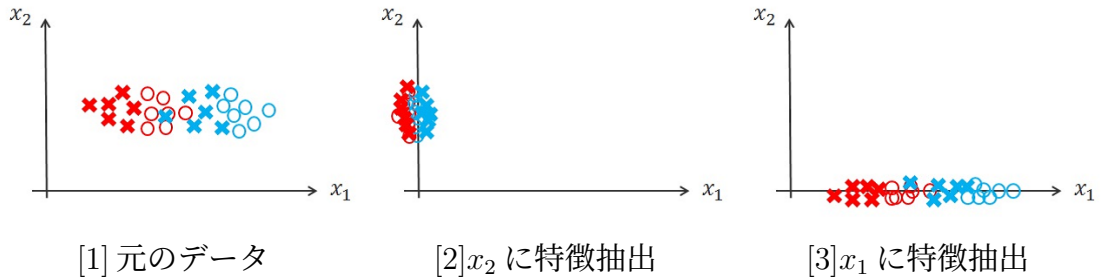


図 9 分散を考慮した特徴抽出の例

ここで分散の保存に関して例を挙げる。図 9[1] のように x_1 が大きいとマル、小さいとバツの二値をもつ赤と青の二つのデータセットを考える。特徴抽出の際に分散が小さくなるということは何らかの情報が失われるということである。図 9[2] のように、分散の小さい方向である x_2 方向に特徴抽出を行うと、マルとバツの判別がつかなくなる。つまり、元のデータに含まれていたマルとバツの情報が失われる。一方で、図 9[3] のように、分散の小さい方向である x_1 方向に特徴抽出を行うと、抽出した特徴量にマルとバツの情報を保存することが出来る。以上の

ように、特徴抽出の際は共通性のみならず、重要な情報を失わないように特徴抽出を行う必要がある。

以上の要件を満たした特徴量を抽出するため、転移成分分析を利用する。転移成分分析とは、二つのデータセットに共通する情報を抽出するのに用いられる手法で、データの持つ分散や、他のデータとの相関を考慮して特徴抽出を行うこともできる。また、転移成分分析を用いる際に画像を汎用特徴量としてベクトル化しておく必要がある。汎用特徴量として、回転や拡大・縮小に頑健なことから、SIFT 特徴量 [12] がよく用いられる。そのため、汎用特徴量として SIFT 特徴量を利用したい。しかし、SIFT 特徴量は画像に応じて得られる特徴量の数が異なるため、特徴ベクトルの大きさが画像によって異なる。転移成分分析を利用するには、画像を一定サイズの特徴ベクトルに変換する必要がある。そのため、汎用特徴量抽出に、一定サイズの特徴ベクトルを抽出できる空間的ピラミッドマッチング [13] を利用する。空間的ピラミッドマッチングでは、画像中のオブジェクトの位置情報など様々な情報を保存して特徴ベクトルを抽出することが出来る。また、方策学習法として EM 方策探索 [14] を利用することで、実環境での学習なしに方策転移を行う。

以下、3.1 節では強化学習の概要について説明する。3.2 節では強化学習に利用する EM 方策探索について説明する。3.2 節では汎用特徴量抽出に利用する空間的ピラミッドマッチングについて説明する。3.3 節ではシミュレーションと実環境に共通する特徴量の抽出に利用する転移成分分析について説明する。

3. 準備

3.1 強化学習概要

強化学習は、行動のルールである方策、状態、状態を別の状態に遷移させる行動、状態の評価値である報酬の4つの要素を用いて、報酬を最大化する方策を求める。本研究では、状態 s_n 、行動 a_n 、方策 $\pi(a_n|s_n)$ とする。このとき状態 s_n と行動 a_n の時系列で表されるエピソード d と、 d の生成確率 $P_\pi(d)$ はステップ数を T としたとき、次式 (1), (2) で表せる。

$$d \equiv (s_1, a_1, s_2, a_2, \dots, s_T, a_T, s_{T+1}) \quad (1)$$

$$P_\pi(d) \equiv P(s_1) \prod_{n=1}^T \pi(a_n|s_n) P_\pi(s_{n+1}|s_n, a_n) \quad (2)$$

このとき、エピソード d での報酬和（リターン） $R(d)$ と、期待報酬 $J(\pi)$ は割引率 γ を用いると次のように表せる。

$$R(d) \equiv \sum_{n=1}^T \gamma^{n-1} R(s_n, a_n, s_{n+1}) \quad (3)$$

$$J(\pi) = \int R(d) P_\pi(d) dd \quad (4)$$

方策に注目した強化学習である方策探索では、期待報酬値 $J(\pi)$ を最大化する方策 $\pi(a_n|s_n)$ を求める。

3.2 EM 方策探索

方策の更新が L 回行われる場合を考える。方策のパラメータを θ とする。期待報酬 $J(\theta)$ は非線形であることが多く、 $J(\theta)$ を最大化する θ を直接求めることは困難である。そこで、EM 方策探索を利用する。EM 方策探索では、期待報酬 $J(\theta)$ ではなく θ に関する容易に最大化できる下界 $J_L(\theta)$ の計算と、下界の更新に

よる θ_{L+1} の更新を交互に繰り返すことで、最終的に $J(\theta)$ を最大化する方策のパラメータ θ^* を求める． $\mathcal{R}(d)$ は非負であると仮定すると、 θ_L を現在のパラメータとして、Jensen の不等式により期待報酬の下界に関する式が求まる．

$$\log J(\theta) \geq \int \frac{\mathcal{R}(d)P(d; \theta_L)}{J(\theta)_L} \log \frac{P(d; \theta)}{P(d; \theta_L)} dd + \log J(\theta_L) \equiv \log J_L(\theta) \quad (5)$$

$$\theta_{L+1} \equiv \arg \max_{\theta} J_L(\theta) \quad (6)$$

この下界の計算は E-step と呼ばれ、 θ に関する最大化問題を解き θ を更新する式が M-step と呼ばれる．また、下界 J_L は θ_L において期待報酬と接するので、各反復において期待報酬値は減少しない．

$$J(\theta_{L+1}) \geq J(\theta_L) \quad (7)$$

パラメータ $J(\theta) = (w, \sigma)$ の更新式は以下のように導出される．

$$\begin{aligned} \mathbf{w}_{L+1} = & \left(\frac{1}{M} \sum_{m=1}^M \mathcal{R}(\mathbf{d}_m^L) \frac{1}{N} \sum_{n=1}^N \phi(\mathbf{s}_{m,n}^L) \phi(\mathbf{s}_{m,n}^L)^T \right)^{-1} \\ & \left(\frac{1}{M} \sum_{m=1}^M \mathcal{R}(\mathbf{d}_m^L) \frac{1}{N} \sum_{n=1}^N \mathbf{a}_{m,n}^L \phi(\mathbf{s}_{m,n}^L) \right) \end{aligned} \quad (8)$$

$$\begin{aligned} \sigma_{L+1}^2 = & \left(\frac{1}{M} \sum_{m=1}^M \mathcal{R}(\mathbf{d}_m^L) \right)^{-1} \\ & \left(\frac{1}{M} \sum_{m=1}^M \mathcal{R}(\mathbf{d}_m^L) \frac{1}{N} \sum_{n=1}^N \left(\mathbf{a}_{m,n}^L - \mathbf{w}_{L+1}^T \phi(\mathbf{s}_{m,n}^L) \right)^2 \right) \end{aligned} \quad (9)$$

ここで、本研究では方策は次式のガウス分布で与えられる．

$$\pi(a|s; \theta) \equiv \frac{1}{\sigma \sqrt{2\pi}} \exp \left(-\frac{(a - k^T \phi(s))^2}{2\sigma^2} \right) \quad (10)$$

パラメータ θ は σ^2 と k である． $\phi(s)$ は既定関数であり，本研究では画像から抽出される特徴量である． EM 方策探索を用いて上記パラメータを更新することで，報酬を最大にする行動を出力する方策を学習する．

3.3 空間的ピラミッドマッチング (Spatial Pyramid Matching:SPM)

画像を汎用特徴量としてベクトル化するために，SPM を利用する． X を画像， Y を複数の画像から抽出された Bag-of-Features とする． 特徴ベクトル群をクラスタリングして画像を分解レベル $0, 1, \dots, L$ のように漸進的にグリッドに分解することで，分解レベル ℓ において画像は 4^ℓ のセルに分解される． 各セルにおいて画像の回転や拡大・縮小に頑健な SIFT 特徴を抽出することで，特徴ベクトルが得られる． SIFT 特徴抽出により得られる特徴ベクトルのサイズは画像により異なるため，SPM により一定サイズのベクトルに変換する． SPM では， X の特徴ベクトルを Y に含まれる M 種類の離散値に量子化し，同じ種類に分類された特徴が X, Y 間で一致する特徴として扱う． ある離散値 $m \in M$ を考えるとき， X, Y に関する特徴量のうち，離散値 m に量子化される特徴量の位置を二次元ベクトル X_m, Y_m で表すと，式 (12) の Pyramid Matching Kernel[15]，式 (13) の histogram intersection function[16] から， X から抽出した特徴ベクトルが式 (11) の Spatial Pyramid Matching Kernel として得られる．

$$K^L(X, Y) = \sum_{m=1}^M \kappa^L(X_m, Y_m) \quad (11)$$

$$\begin{aligned} \kappa^L(X, Y) &= \mathcal{I}^L + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} (\mathcal{I}^\ell - \mathcal{I}^{\ell+1}) \\ &= \frac{1}{2^L} \mathcal{I}^0 + \sum_{\ell=1}^L \frac{1}{2^{L-\ell+1}} \mathcal{I}^\ell \end{aligned} \quad (12)$$

$$\mathcal{I}(H_X^\ell, H_Y^\ell) = \sum_{i=1}^D \min(H_X^\ell(i), H_Y^\ell(i)) \quad (13)$$

H_X^ℓ, H_Y^ℓ はそれぞれ分解レベル ℓ における X, Y の SIFT 特徴のヒストグラムである。式 (12) において, $\mathcal{I}(H_X^\ell, H_Y^\ell)$ は $\mathcal{I}^\ell$ のように省略されている。また, $H_X^\ell(i), H_Y^\ell(i)$ はそれぞれ, 分解レベル ℓ において特徴空間がセルに分解された際に, i 番目のセル内にある X, Y の特徴量の数である。

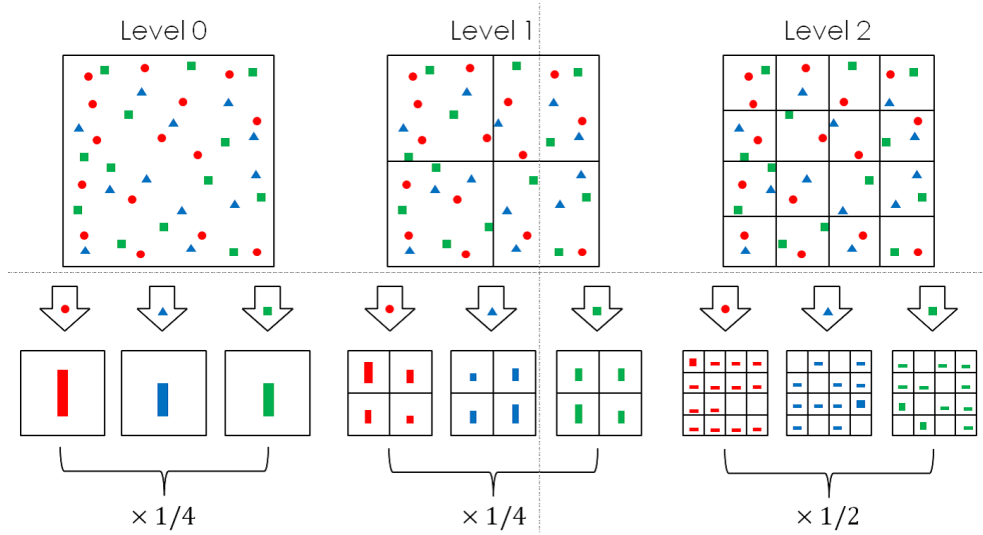


図 10 分解レベル 2 の SPM による特徴抽出の一例。例では画像は丸, 三角, 四角の三つの特徴に離散化されている。各レベルにの各グリッドにおいてそれぞれの特徴量を数え上げ, ヒストグラムとして表現する

3.4 転移成分分析

シミュレーションで学習した方策を実環境で利用するという問題設定は, 転移学習問題と捉えることが出来る。転移学習とは, ある領域で学習したモデルを別の領域に利用するための技術である。本研究では, 学習元の領域であるソースドメインと, 転移先の領域であるターゲットドメインの二つの領域を扱う。ソースドメインから入力データセット $X_S = \{x_{S1}, \dots, x_{Sn_1}\}$ と出力データセット $Y_S = \{y_{S1}, \dots, y_{Sn_1}\}$, ターゲットドメインから入力データセット $X_T = \{x_{T1}, \dots, x_{Tn_2}\}$ が観測されたとする。ここで, $x_S, x_T \in \mathcal{X}$, $y_S \in \mathcal{Y}$ と仮定する。

転移学習の目的は、ターゲットドメインの入力に対する出力を正しく予測するモデルを作成することである。ただし、ターゲットドメインの出力データは与えられないため、ソースドメインの入出力データを上手く活用してターゲットドメインの出力を予測するモデルを作成しなければならない。

転移成分分析では、 $p(x_S) \neq p(x_T)$ のとき、 X_S および X_T を用いて $p(\phi(x_S)) \approx p(\phi(x_T))$ かつ $p(y_S|\phi(x_S)) \approx p(y_T|\phi(x_T))$ を満たす特徴量 ϕ を見つける。つまり、入力の分布は異なるものの、入力と出力の対応がソースドメインとターゲットドメインで一致すると考えられる場合に、各ドメインの入力が同じような分布になる特徴量 ϕ に変換する。これにより、 ϕ を用いて、 $\phi(X_S)$ と Y_S を使って学習される回帰モデルを用いることで、 ϕ の上でターゲットドメインの出力の予測を行うことが出来る。

以下、転移成分分析において、転移の指標として用いられる指標である MMD(Maximum Mean Discrepancy)[17] および HSIC(Hilbert Schmidt Independence Criterion)[18] について簡単に説明する。その後、これらに基づく転移成分分析のアルゴリズムを示す。

3.4.1 Maximum Mean Discrepancy

MMD は二つの変数 $x, y \in \mathcal{X}$ について、確率分布 $p(x)$ および $p(y)$ の距離を、カーネル関数に備わる RKHS (Reproducing Kernel Hilbert Space) 上でのデータセット X および Y の距離によって近似的に推定する手法である。

カーネル関数に備わる特徴量 ϕ を用いて、MMD の推定量は以下の式で与えられる。

$$\text{MMD}(X, Y) = \left\| \frac{1}{n_1} \sum_{i=1}^{n_1} \phi(x_i) - \frac{1}{n_2} \sum_{i=1}^{n_2} \phi(y_i) \right\|_{\mathcal{H}} \quad (14)$$

ここで、 $\|\cdot\|_{\mathcal{H}}$ は RKHS ノルムである。つまり、MMD は、RKHS 上でのデータセットの平均値を比較することで、二つの確率分布の中心の距離を比較している。MMD は $p(x) = p(y)$ のとき、漸近的に 0 に近づく。MMD が小さいほど各データセットの分布が近づき、 X と Y は共通性の高いデータとなる [19]。

3.4.2 Hilbert-Schmidt Independence Criterion

HSICはRKHS上で相互共分散作用素のHilbert-Schmidt normを用いて、二つの確率変数 $x \in \mathcal{X}$ と $z \in \mathcal{Z}$ の独立性を、データセット X および Z を用いて推定する手法である。カーネル関数 $k(x, x')$, $k(z, z')$ を定義すると、HSICは以下の式で与えられる。

$$\text{HSIC}(x, z) = \frac{1}{(n-1)^2} \text{tr}(H K_x H K_z) \quad (15)$$

また、本研究では特に言及がない限り、カーネル関数は以下のRBFカーネルで与えられる。

$$k(x, z) = \exp\left(\frac{-\|x - z\|^2}{2}\right) \quad (16)$$

ここで、 K_x, K_z はそれぞれ各要素が X や Z のカーネル関数であるカーネルグラム行列である。また $H = I - \frac{1}{n} l l^T$ である。 l は要素が全て1から成るベクトルである。 n は x と z のデータ数である。HSICは x と z が独立であれば漸近的に0に近づく。反対に、 x と z に強い相関がある場合、HSICは大きな値となる [20]。

3.4.3 転移成分分析

転移成分分析の目的は、MMDを利用して、以下の二つの要求を満たすように特徴量 ϕ を見つける事である。

- a $p(\phi(x_S))$ と $p(\phi(x_T))$ の差異が少ない
- b $\phi(x_S)$ と $\phi(x_T)$ が x_S, x_T の大事な特徴を保存

ただし、 ϕ は強い非線形である場合が多く、直接見つけることは難しいため、パラメトリックカーネルマップモデル [21] を利用し、入力の特徴空間へ写像するパラメータ W を見つける問題へと簡略化する。

まず、 X_S と X_T を用いて以下のカーネルグラム行列を求める。

$$K = \begin{bmatrix} K_{SS} & K_{ST} \\ K_{TS} & K_{TT} \end{bmatrix} \quad (17)$$

$K_{SS}, K_{TT}, K_{ST}, K_{TS}$ はそれぞれ x_S や x_T の RKHS 上での内積であり、カーネル関数 k を用いて計算される。ここでは、Maximum Mean Discrepancy Embedding(MMDE)[20] を基本として、 $K = (KK^{-1/2})(K^{-1/2}K)$ という分解表現を考え、これを $\tilde{W}$ を利用して特徴空間へ写像する。

$$\tilde{K} = \left(KK^{-1/2}\tilde{W}\right) \left(\tilde{W}^TK^{-1/2}K\right) = KWW^TK \quad (18)$$

ここで、 $W = K^{-1/2}\tilde{W}$ である。これにより、 x_i, x_j のカーネル値と特徴量 $\phi(x_i)$ は、 W に依存する以下の式で表される。

$$\tilde{k}(x_i, x_j) = k_{x_i}^T WW^T k_{x_j} \quad (19)$$

$$\phi(x_i) = W^T k_{x_i} \quad (20)$$

ここで、 $k_x = [k(x_1, x), \dots, k(x_{n_1+n_2}, x)]^T$ である。また、 n_1, n_2 はそれぞれ X_S, X_T のデータ数である。 $\tilde{K}$ を用いることで ϕ 上における X_S と X_T の距離は MMD を用いて以下の式で与えられる。

$$\text{Dist}(X_S, X_T; W) = \text{tr}((KWW^TK)L) \quad (21)$$

$$= \text{tr}(W^TKLKW) \quad (22)$$

L は $x_i, x_j \in x_S$ のとき $L_{ij} = \frac{1}{n_1^2}$, $x_i, x_j \in x_T$ のとき $L_{ij} = \frac{1}{n_2^2}$, その他は $L_{ij} = \frac{-1}{n_1 n_2}$ となる行列である。

また、要求 b の基準として、RKHS 上でのソース・ターゲットドメインの入力分散を採用すると、転移成分分析は以下の最適化問題として定式化される。

$$\min_W \quad \text{tr}(W^TKLKW) + \mu \text{tr}(W^TW) \quad (23)$$

$$\text{s.t.} \quad W^TKHKW = I_m \quad (24)$$

ここで、 $\mu > 0$ は正則化項の重み、 I_m は単位行列である。式 (24) の拘束条件のものと式 (23) の最適化問題が TCA である。

一方、要求 b の基準として、RKHS 上における別の変数 z との依存性を採用すると、HSIC を拘束条件として、以下の最適化問題に帰着される。

$$\begin{aligned}
& \min_W \quad \text{tr}(W^T K L K W) + \mu \text{tr}(W^T W) \\
& \text{s.t.} \quad W^T K H K_z H K W = I
\end{aligned} \tag{25}$$

K_z は変数 z から成るカーネルグラム行列である．この式 (25) の拘束条件のものと最適化問題は，半教師あり転移成分分析 (Semi-supervised TCA: SS-TCA) の一種とみなせる．

TCA, SS-TCA のどちらも固有値問題に帰着される．結果として求められる固有値の上位 M 個の固有ベクトルから成る行列が，各データを特徴空間へ写像する最適な W^* となる [11]．

4. 提案法：方策転移学習

本章では，3章で説明した要素技術を用いて，シミュレーションにおいて強化学習により獲得された方策を，見かけ上の誤差を伴う実環境に転移させる枠組みとして方策転移学習を提案する．また，特徴量の抽出基準について二つの基準を示すことで，2種類の方策転移法を提案する．

4.1 分散を基準とした特徴量による方策転移学習概要

分散を基準とした方策転移学習の処理手順は以下の通りである．また，その概要を図 11 に示す．

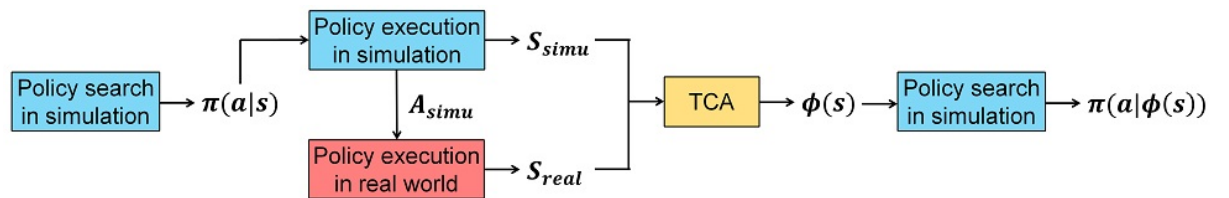


図 11 分散を基準とした特徴量による方策転移学習概要

- 1 シミュレーションにおいて，EM 方策探索を用いて強化学習を適用して，報酬を最大化する方策 $\pi(a|s)$ を探索する．
- 2 シミュレーションにおいて方策 $\pi(a|s)$ を実行し，状態および行動系列 S_{simu} ， A_{simu} を生成する．この際，方策の学習は行わない．
- 3 行動系列 A_{simu} を実環境において実行し，状態系列 S_{real} を生成する
- 4 S_{simu} ， S_{real} を用いて，TCA を適用することで， $\phi(S_{simu})$ と $\phi(S_{real})$ の類似性を考慮した ϕ を抽出する．
- 5 シミュレーションにおいて $\pi(a|\phi(s))$ を用いて方策探索を実行し， ϕ を入力とする最適な方策を学習することで，実環境に転移できる方策を得る．

本手法では、特徴抽出の際に、 $\mathcal{S}_{\text{simu}}$, $\mathcal{S}_{\text{real}}$ の RKHS 上での分散を保存するように ϕ を抽出することで、特徴抽出の際に失われる情報を抑えつつ、共通の特徴を抽出することが出来る．以下本手法を、TCA を用いた提案法と表記する．

4.2 報酬値を基準とした特徴量による方策転移学習概要

強化学習における報酬値を基準とした方策転移学習の概要を図 12 に示す．

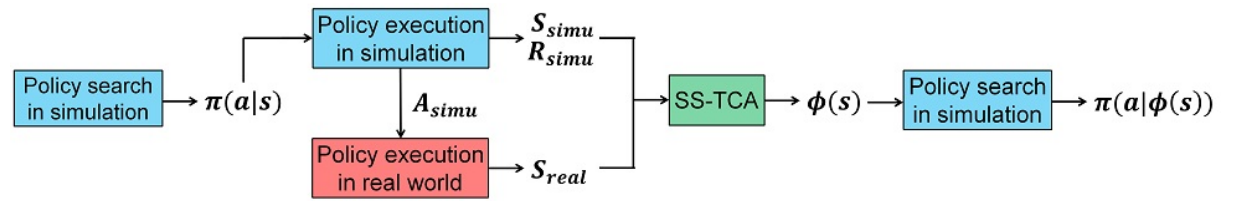


図 12 報酬値を基準とした特徴量による方策転移学習概要

方策転移学習の手順は分散を用いた場合とほぼ同等であるが、手順 4 のみ異なる．本手法では、手順 4 において、 $\mathcal{S}_{\text{simu}}$, $\mathcal{R}_{\text{simu}}$, $\mathcal{S}_{\text{real}}$ を用いて、SS-TCA を適用する．これにより、 $\phi(\mathcal{S}_{\text{simu}})$ と $\phi(\mathcal{S}_{\text{real}})$ の類似性と、 $\phi(\mathcal{S}_{\text{simu}})$ と $\mathcal{R}_{\text{simu}}$ の依存性の両方を考慮した特徴量 ϕ を抽出する．

上記基準について、報酬値 $\mathcal{R}_{\text{simu}}$ の利用は、強化学習の状態特徴抽出問題に関する Hachiya らの研究 [22] に基づいている．状態の分散などをもとに特徴を選択すると、後の高い報酬に関連する情報を失う恐れがある．そこで、Hachiya らは以下の基準を提案した．

$$R_n \perp s_n \mid \phi(s_n) \quad \forall n \quad (26)$$

この基準では、報酬値 R_n と状態変数 s_n が独立とみなせるような特徴量 $\phi(s_n)$ が良い特徴ということになる．以下本手法を、SS-TCA を用いた提案法と表記する．

5. 実験

本章では提案した手法を使って、学習した方策を見かけ上の誤差の存在する環境に転移できるかをシミュレーションと実環境にて検証する．5.1 節では，まずシミュレーションで学習した方策を，見かけ上の誤差を含むシミュレーション環境に直接転移し，方策の性能劣化が起こることを確認する．次いで，TCA を用いた提案法や SS-TCA を用いた提案法により方策転移を行い，各手法の有効性を確認する．5.2 節では，シミュレーションで学習した方策を実環境に転移し，見かけ上の誤差を多分に含む環境での各手法の有効性を確認する．

5.1 見かけ上の誤差を含むシミュレーション環境に方策転移

5.1.1 実験システム

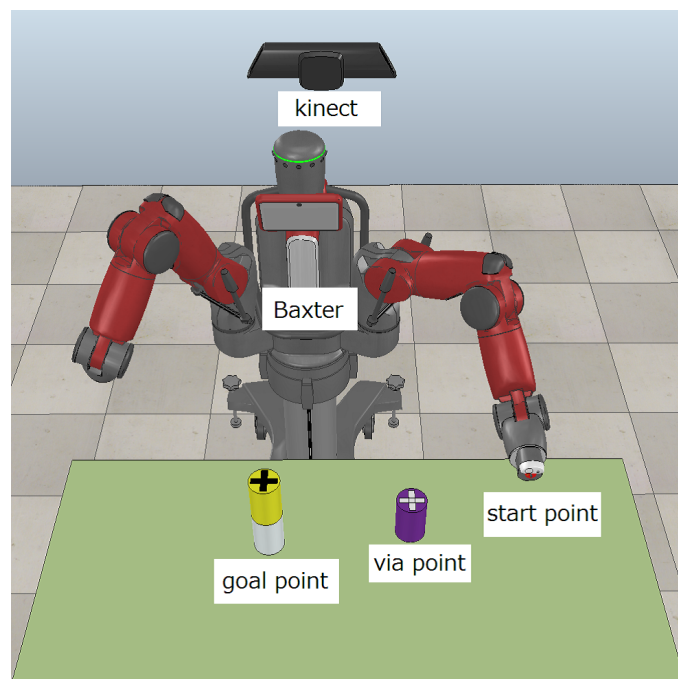


図 13 方策学習環境全体図

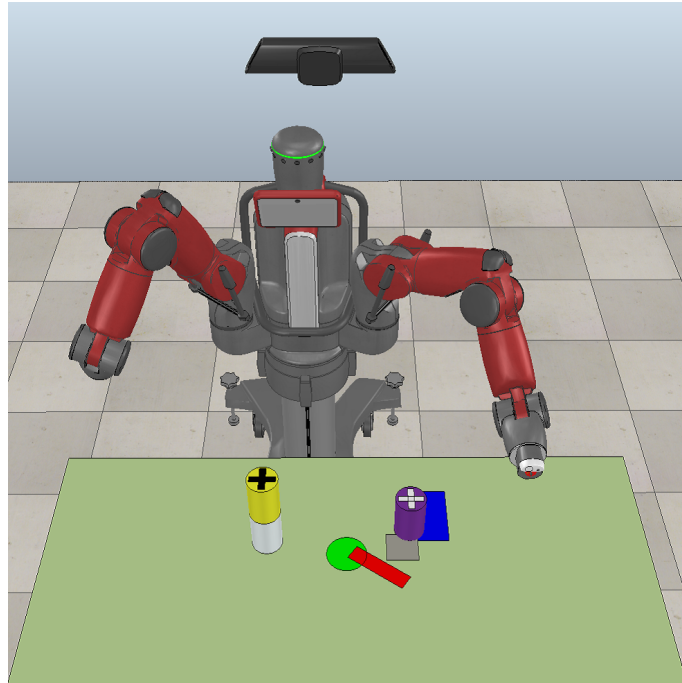
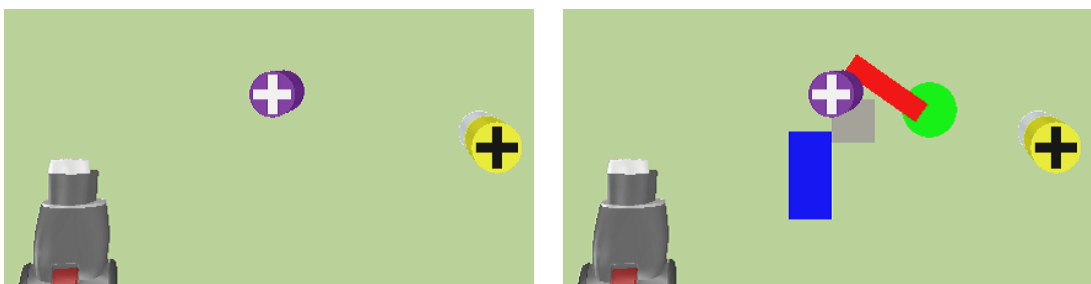


図 14 方策転移先の環境全体図



(a) 方策学習環境俯瞰図

(b) 方策転移先の環境俯瞰図

図 15 シミュレーション環境俯瞰図

方策学習環境全体図と、方策を転移する見かけ上の誤差を伴う環境全体図を図 13, 14 に示す。また、図 13 の kinect から得られる両環境の俯瞰図を図 15 に示す。本実験では、シミュレーション環境として、Coppelia Robotics 社の V-REP を用いる。システムは V-REP 上の双腕ロボット Baxter, kinect から成る。図 13(a) の環境において、Baxter の左手先を図中の経由地点上を通過させ、目標地点へ近づけるタスクを実行する。経由地点は図 16 に示す上下 2 種類とした。

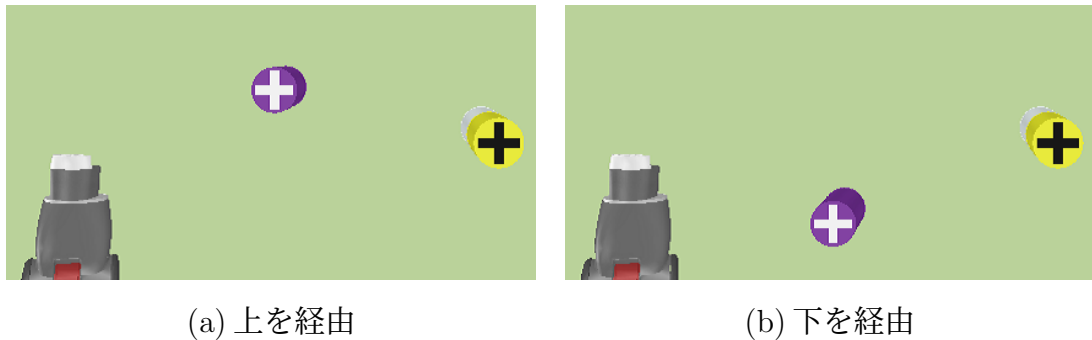


図 16 二種類の経由地点

システム全体の処理手順は以下の繰り返しで動作する。

1. シミュレーション内の kinect から ROS(Robot Operating System)[23] を利用して画像を取得する
2. 取得した画像に対し、SPM を用いて特徴量を抽出する
3. 特徴量を方策に直接入力（直接転移）、あるいは TCA か SS-TCA を用いて共通の特徴量に変換し方策に入力（提案法）することで、Baxter の手先の移動先の座標を生成する
4. 生成した座標を ROS を利用して V-REP 上の Baxter に送信し、手先を移動させる

5.1.2 実験設定

初めに、強化学習に利用される行動 a_n 、状態 s_n 、報酬 r_n 、報酬和 R_n に関して説明する。図 16 はシミュレーションの初期状態であり、Baxter の手先は、図 16 の横方向を x 、縦方向を y とすると、 $(0.700, 0.803)$ の位置にある。また目標地点は $(-0.100, 803)$ 、経由地点は上下 2 種類でそれぞれ、 $(0.350, 0.943)$ 、 $(0.350, 0.663)$ である。Baxter の手先の可動範囲は、 x 方向が $[0.000 : 0.700]$ 、 y 方向が $[0.600 : 1.000]$ で制限される。行動 a_n は図 16 における Baxter の手先の y 方向の動きとし、状態 s_n は Baxter の手先の y 方向の位置とする。また、行動 a_n は $[-0.070 : 0.070]$ の範囲で制限される。Baxter は前述のとおり、左手先を図中の経由地点上を通過させ、目標地点へ近づけるタスクを実行するが、 x 方向の移動は 1 ステップごとに -0.077 移動する。以上より、状態遷移は次式で表される。

$$s_{n+1} = s_n + a_n \quad (27)$$

報酬 r_n は Baxter の手先位置と、経由地点や目標地点までのユークリッド距離を考慮した次式で与えられる。

$$r_n = \begin{cases} 0.4 \times (t) \times (0.3 - |y_v - y|) & (x \leq x_v) \\ 20 \times \exp\left(\frac{-(150 \times (0.8030 - y)^2)}{2}\right) & (t = T) \end{cases}$$

ここで、 x, y はロボットの手先の x, y 座標、 x_v, y_v は経由地点の x, y 座標、 t は現在のステップ数とし、最大ステップ数 T は 10 ステップである。エピソード d における報酬和 R_n は、式 (3) で定義される。学習率 γ は 0.99 に設定した。学習はエピソード数を 100、ステップ数を 10 に設定して行った。上記の設定で方策を 3 つ学習しておき、それぞれを提案手法を用いて転移した。

次に、方策転移の手順と、方策に入力する特徴量の生成に関して説明する。見かけ上の誤差を含むシミュレーション環境への方策転移は以下の手順で行った。

- 1 図 15(a) の環境において報酬を最大化する方策 $\pi(a|s)$ を事前に学習しておき、その方策を実行することで行動系列 $\mathcal{A}_{\text{simu}}$ と対応する状態系列 $\mathcal{S}_{\text{simu}}$ 、報酬系列 $\mathcal{R}_{\text{simu}}$ を生成する。

- 2 図 15(b) の環境で $\mathcal{A}_{\text{simu}}$ を再現し, 状態系列 $\mathcal{S}_{\text{error}}$ を生成する.
- 3 $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{error}}$ に, 3.4.3 節で述べた TCA や SS-TCA を行うことで, 各データを特徴空間に写像する W を得る. 式 (19,20) より, 特徴量 $\phi(\mathcal{S}_{\text{simu}})$, $\phi(\mathcal{S}_{\text{error}})$ を生成する.
- 4 $\phi(\mathcal{S}_{\text{simu}})$ を利用して方策探索を行うことで, 見かけ上の誤差を含む画像を入力しても動作する方策 $\pi(a|\phi(s))$ を生成する.

方策転移の手順 2 における, $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{error}}$ は 20 エピソード分のデータから生成した. また, $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{error}}$ の対応の例を図 17, 18 に示す. 赤線は手先の軌道である.

特徴量の生成に関して, 初めに kinect から取得される画像から図 16 の様な 430×223 ピクセルの画像を切り出す. 方策に入力する特徴量は, 直接転移の場合は上記画像に SPM を用いて抽出した特徴量, 提案手法を用いる場合は上記画像に SPM を適用した後に TCA や SS-TCA を用いて抽出した特徴量となる. SPM のパラメータである分解レベル ℓ は 2 とし, 量子化数 M は, $M = 300$ とした. また, 各分解レベルにおけるヒストグラムを足し合わせる際の重みは, 分解レベル 0 から 2 までそれぞれ, $\{\frac{1}{4}, \frac{1}{4}, \frac{1}{2}\}$ とした. 以上から, 画像は 21 の小領域に分解され, 各領域から 1×300 次元の特徴量が抽出されることから, SPM により画像 1 枚から 1×6300 次元の特徴量が得られる. TCA のパラメータに関して, 重み μ は 1 とし, 式 (23) から求められる W の次元は 50 次元に設定した.

転移した各方策に関して 10 エピソード分の平均報酬値の平均と標準偏差により, TCA と SS-TCA を用いた提案手法の有効性を評価する. また比較として, TCA や SS-TCA の代わりに, 主成分分析 (Principal Component Analysis: PCA) [24] を用いて方策転移を行った場合や, 元の方策 $\pi(a|s)$ に図 15(b) の環境の画像を入力する直接転移についても評価を行う.

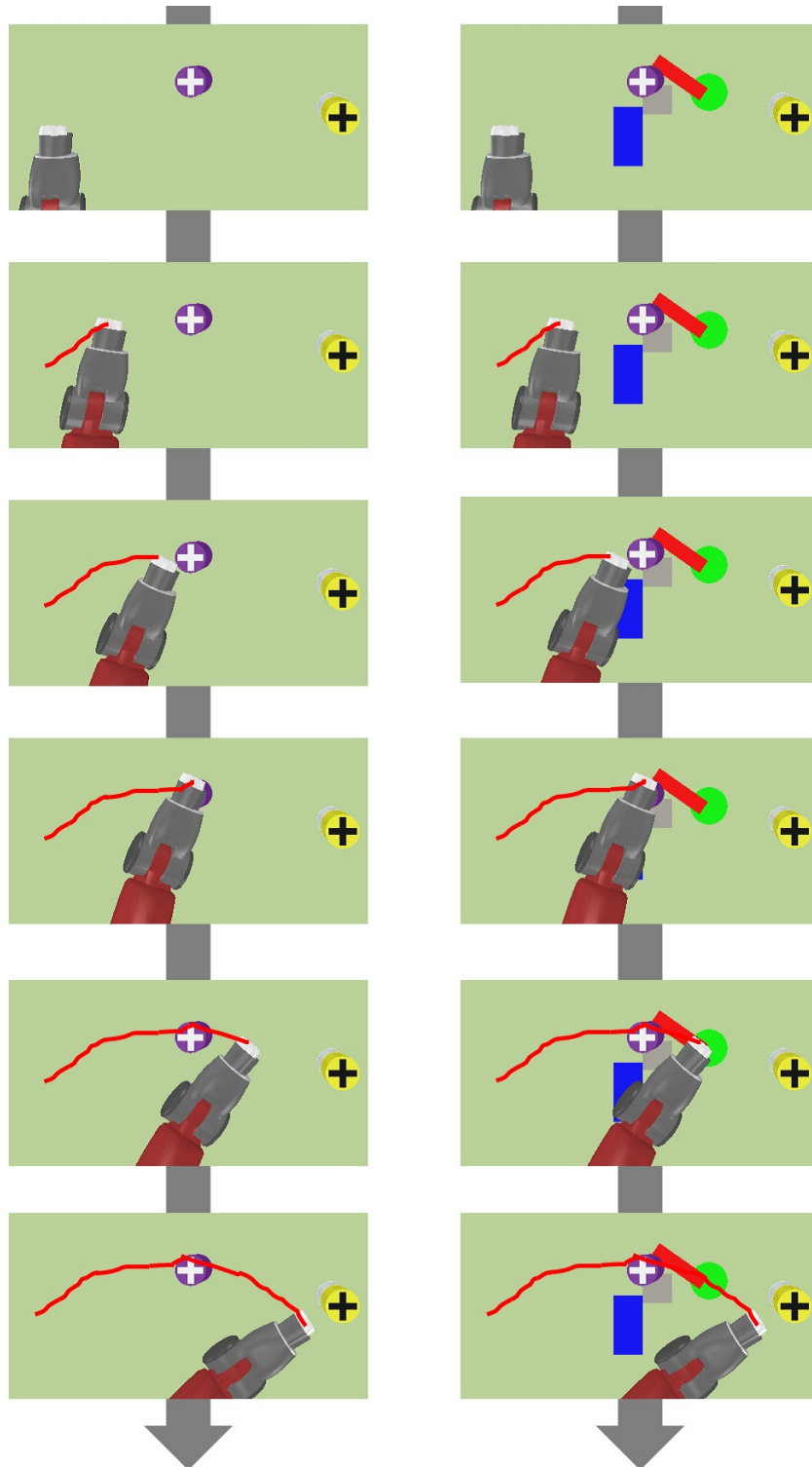


図 17 手順2における, $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{error}}$ の対応の例 (上)

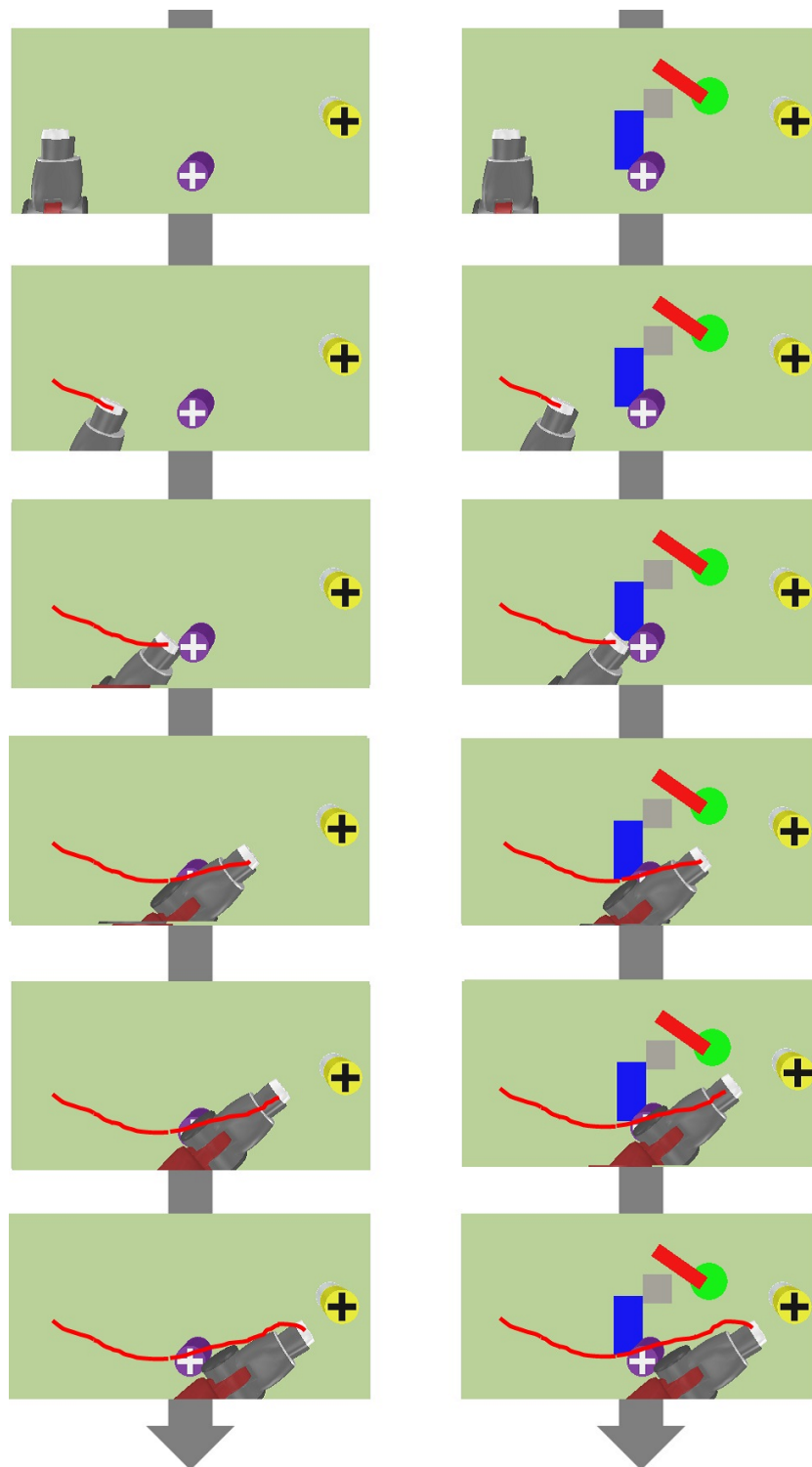


図 18 手順 2 における, $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{error}}$ の対応の例 (下)

5.1.3 実験結果

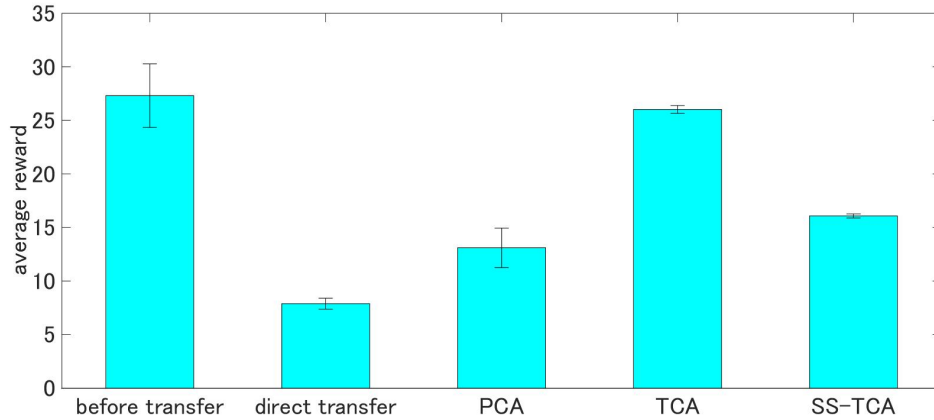


図 19 実験結果 (シミュレーション)

直接転移, PCA, TCA, SS-TCA を用いて方策転移を行い生成した方策 $\pi(a_n|\phi(s_n))$ に図 15(b) の環境の画像を入力した場合と, 元の方策 $\pi(a|s)$ に 15(a) の環境の画像を入力した場合の結果 (元環境) を図 19 に示す. エラーバーは標準偏差を表す. また, 各手法を用いてタスクを実行した場合の一例を図 20, 21 に示す. 図中の赤線は手先の軌道である. 図 19 の直接転移の結果から, 見かけ上の誤差を含む環境に方策を直接転移した場合, 方策の性能劣化が起こることが確認できた. 図 21 中の直接転移がその一例である. この例では経由地点にある程度手先を近づけることはできたが, 目標地点に手先を運ぶことはできなかった. また, TCA を用いた場合は, 平均報酬値が元環境のものとほぼ等しく, 標準偏差も小さいことから, 方策転移が十分に行われたと言える. 図 20, 21 中の TCA の一例では, 経由地点上を通過し, 目標地点に手先を運ぶことが出来ていることから, 方策転移を行えたことがうかがえる. 一方, SS-TCA や PCA を用いた場合は平均報酬値が直接転移による平均報酬値よりは高いものの, 元環境の平均報酬値より低くなってしまっており, 方策転移が十分に行われたとは言えない. 図 20 中の PCA を見ると, 経由地点上を通過できず, 目標地点に手を近づけることもできていないことから, 方策転移が十分に行えていないことが分かる. SS-TCA に関して

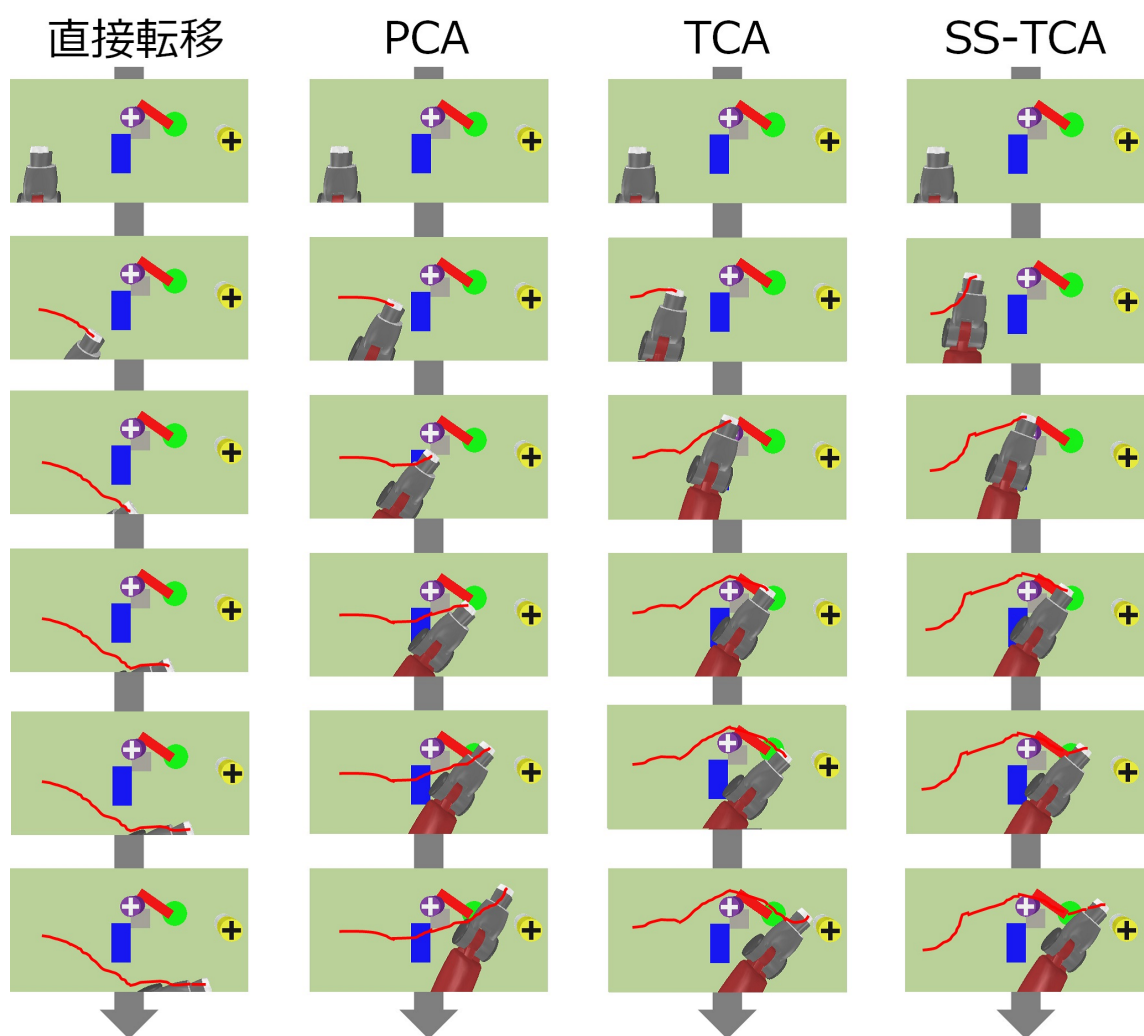


図 20 タスク実行例（上）

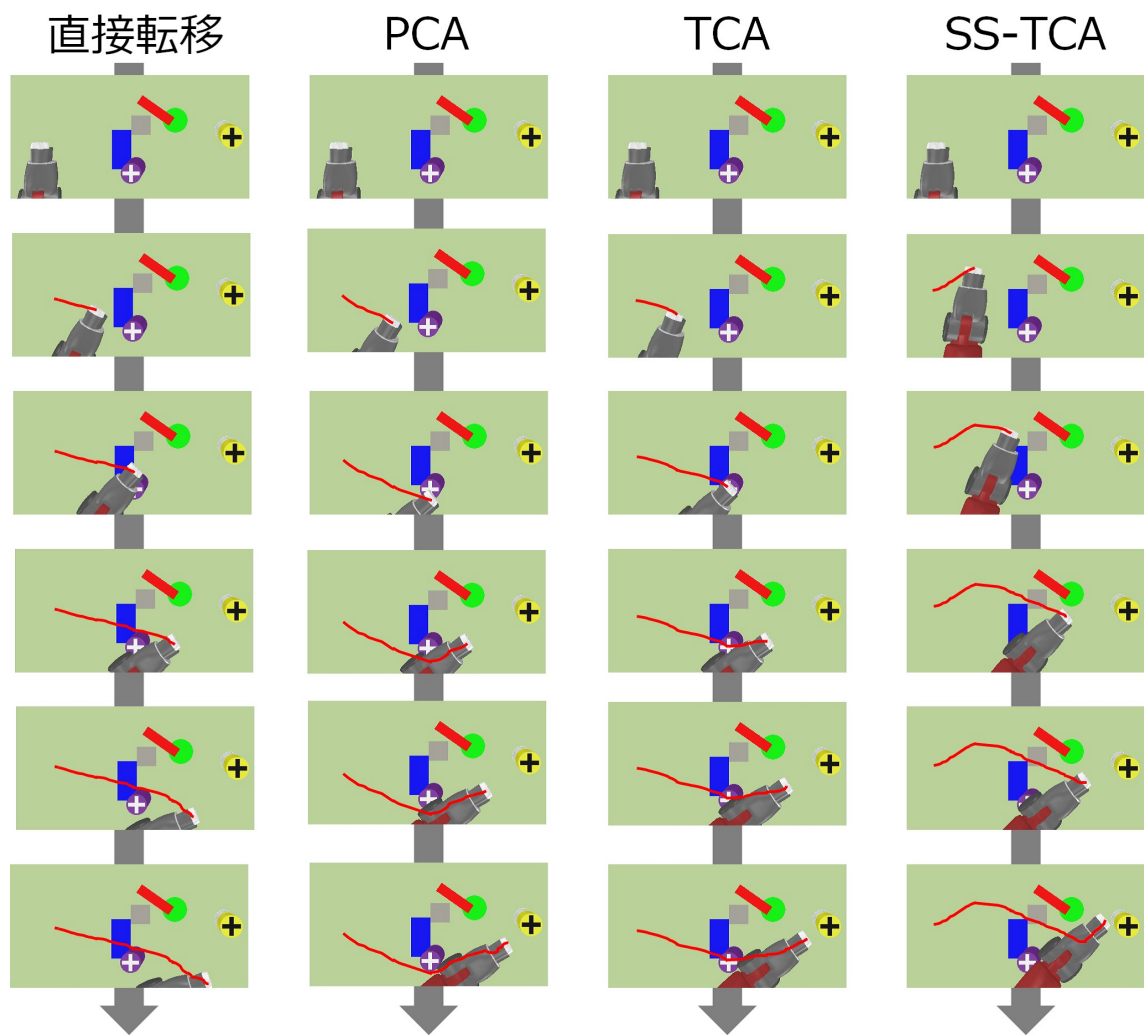


図 21 タスク実行例（下）

言及すると、報酬に関係のある情報を保持するように、両環境に共通する特徴量が抽出されているにもかかわらず、十分には方策転移を行えていない。図 20 中の SS-TCA の例では、経路地点上を通過することはできなかったが、目標地点に手先を近づけることはできていることから、方策転移が完全には行われていないことが分かる。

方策転移が行えていない原因として、2 章で説明した特徴量に関する要件 1 を満たしていない事が原因だと考えられる。つまり、画像から抽出した $\phi(s_{simu})$ と $\phi(s_{error})$ が共通な特徴を含んでおらず、類似度が低いため、方策転移が行えていないと考えられる。そのため、次のような手順で図 17 の様な、対応する二つの環境の画像群の類似度を計測した。

- 1 次のような手順で図 17 の様な、対応する二つの環境の画像群を 1 エピソード分用意する。
- 2 画像群に SPM を適用し、特徴ベクトルを抽出する。
- 3 手順 2 で抽出した特徴ベクトルを直接、あるいは PCA, TCA, SS-TCA を用いて特徴量を抽出し、 $\phi(s_{simu})$ と $\phi(s_{error})$ を生成。
- 4 $\phi(s_{simu})$ と $\phi(s_{error})$ を RBF カーネルに入力することで各特徴量の類似度を計測しグラム行列を生成する。

この手順において、式 16 中の x, z は $\phi(s_{simu})$ と $\phi(s_{error})$ に対応し、両者の類似度が高い場合は 1 に近い値が、低い場合は 0 に近い値が得られる。

手順 1 における 1 エピソード分の画像とそれぞれの類似度を表すグラム行列を図 22, 23, 24 に示す。

図 22 中の番号は図 24 の各図の縦軸の番号と、図 23 中の番号は図 24 の各図の横軸の番号と対応しており、画像から抽出した各特徴量の類似度を表している。明るい色ほど類似度が高く、暗い色ほど類似度が低い。図 24 の naive（直接転移の際の特徴量の類似度）を見ると、図 22(1) と図 23(1) のような同じ姿勢を取っている画像の類似度は高いものの、図 22(1) と図 23(3) のような、手先位置が少し変化しただけのものに関しては類似度が低くなってしまっている。また、図 22(6)

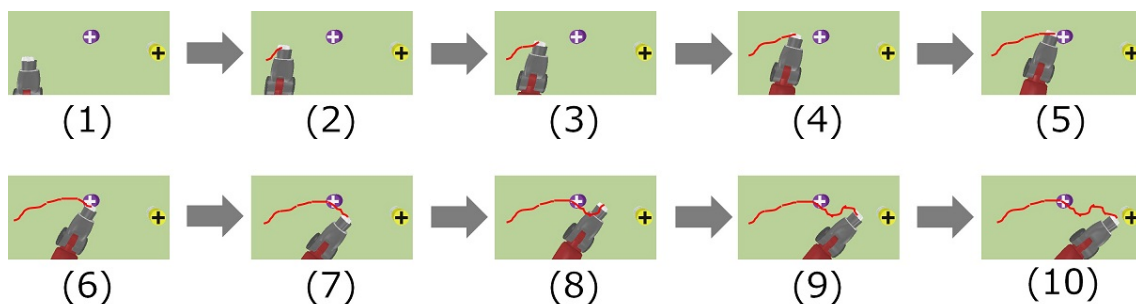


図 22 類似度計測用画像（元環境）

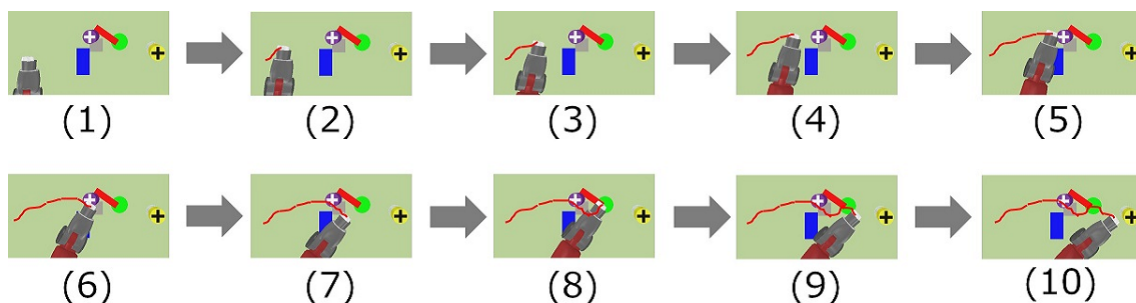


図 23 類似度計測用画像（転移先）

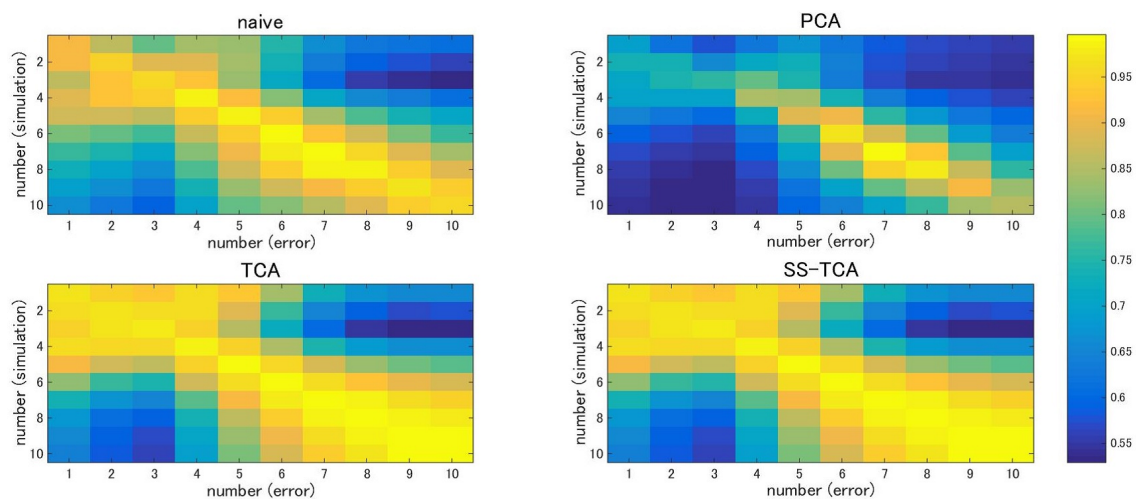


図 24 各手法による特徴量の類似度の比較

と図 23(6) のような、見かけ上の誤差の原因である物が、腕により隠れた画像は、類似度が高くなっている。このことから、見かけ上の誤差が加わると、特徴量の類似度が低くなり差異が大きくなることが分かる。また、図 24 の PCA を見ると、図 22 と図 23 において同じ姿勢の画像であっても類似度が低くなっている。これは PCA がデータの分散が大きくなるよう特徴抽出を行うため、違いが大きくなるように特徴抽出を行ったためである。図 24 の TCA や SS-TCA をみると TCA, SS-TCA で抽出した特徴量に関しては類似度が高い。ただし、図 19 を見ると SS-TCA の場合は方策転移が十分には行えていない。以上から、SS-TCA により方策転移が十分には行えない理由が二つ考えられる。1 つ目は、見かけ上の誤差が加わったことにより、元の環境において報酬と相関の高い特徴が、同様に検出されなくなってしまう、両環境の類似性と特徴量と報酬値の相関を保持した特徴量をうまく抽出できなかったためであると考えられる。2 つ目に、特徴量抽出の際に保持すべき情報が、報酬値と特徴量の相関以外にも存在し、報酬値との相関を保持するだけではその情報が抜け落ちてしまうが、分散を保持するように特徴抽出を行うと、その情報や報酬値と特徴量の相関をいくらか保存できたためと考えられる。このどちらか、あるいは両方の理由から SS-TCA を用いた提案手法は方策転移が十分には行えなかったと考えられる。

以上から、見かけ上の誤差を含む環境に方策転移を行う場合、提案法が有効であることが確認できた。特に TCA を用いた提案法が有効であることが確認できた。一方で、SS-TCA を用いた提案法は完全に方策転移が行えたとは言えない結果が得られた。このことから、二つの環境に共通する特徴量を抽出する際は、報酬値と特徴量の相関を保存するよう特徴抽出するだけでは十分ではなく、二つの環境の分散を保存するよう特徴抽出する方が方策転移を行う上で有効であることが分かった。

5.2 実環境に方策転移

5.2.1 実験システム

実験環境を図 25, 26 に示す. また, kinect から撮影した俯瞰図を図 27 に, 前節のシミュレーションと同様に, システムは Rethink Robotics 社の双腕ロボット Baxterk と kinect から成る. kinect や経由地点, 目標地点のオブジェクトは, シミュレーションと同様のサイズであり, 図 13(a) のシミュレーション環境と同様の位置に設置した. これらは ROS で構成され, ROS ネットワーク上にあるコンピュータにより画像の取得やロボットの制御が行われる. システム全体の処理手順は, 前節と同様である.

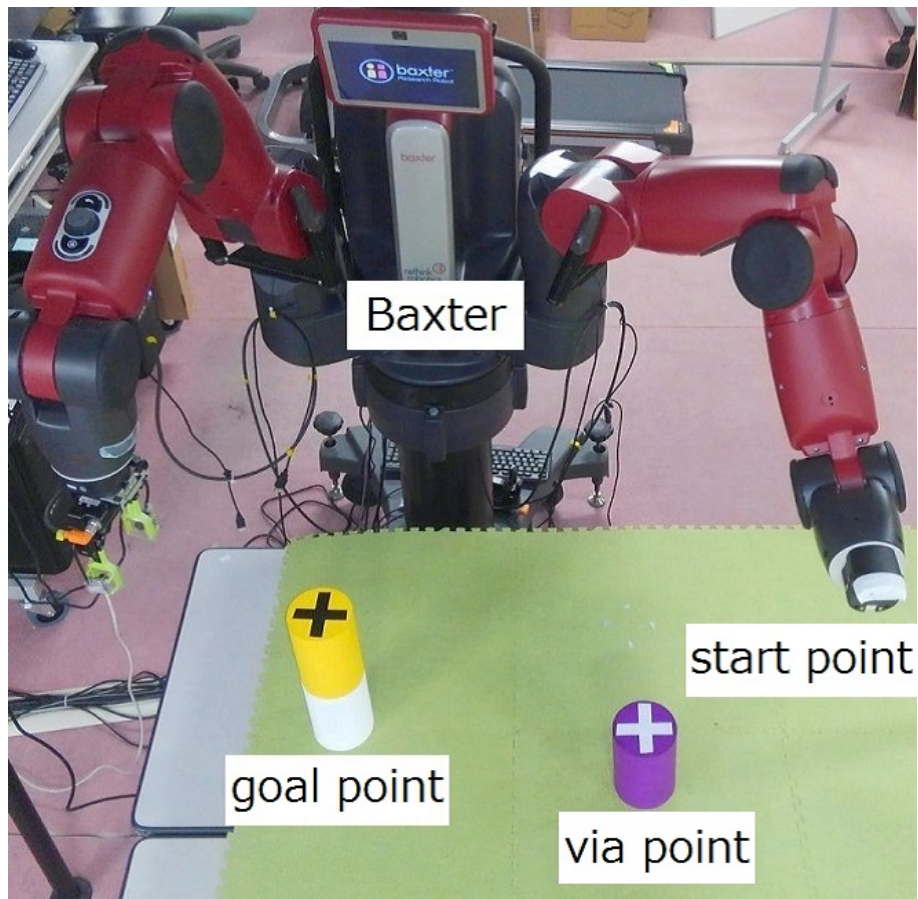


図 25 Baxter



図 26 kinect



図 27 俯瞰図

5.2.2 実験設定

強化学習や特徴量，左手先の可動範囲等に関して前節と同様の設定で方策を 3 つ学習しておき，それぞれを提案手法を用いて転移した．経由地点や目標地点の位置に関して，Baxter の手先をそれぞれの座標まで移動させ，その真下にオブジェクトを設置することで，シミュレーションと同様の位置に設置した．

方策転移の手順に関して，手順は前節と同様であるが， $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{real}}$ は 16 エピソード分のデータから生成した．図 28, 29 に $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{real}}$ の対応の例を示す．赤線は手先の軌道を表す．シミュレータと実際の Baxter の逆運動学の仕様が異なるため，手先位置は概ね同じであるものの，姿勢が異なるようになった．この差異の影響により，方策転移の精度は前節と比べて劣化すると考えられる．ほかのシミュレーションと実環境の差異として，実環境のバクスターの腕にある小さな穴や，マットの継ぎ目や陰影などが挙げられる．

5.1 節と同様に三つの方策 $\pi(a|s)$ に対して提案法を行い，生成された各方策 $\pi(a_n|\phi(s_n))$ に実環境画像から抽出した特徴量を入力した場合の平均報酬値の平均と標準偏差を用いて評価を行った．また，比較として SS-TCA や TCA の代わりに，主成分分析 (PCA) [24] を用いて生成した方策に各画像を入力した場合や，元の方策に各画像を入力する直接転移に関しても 5.1 節と同様に評価を行う．

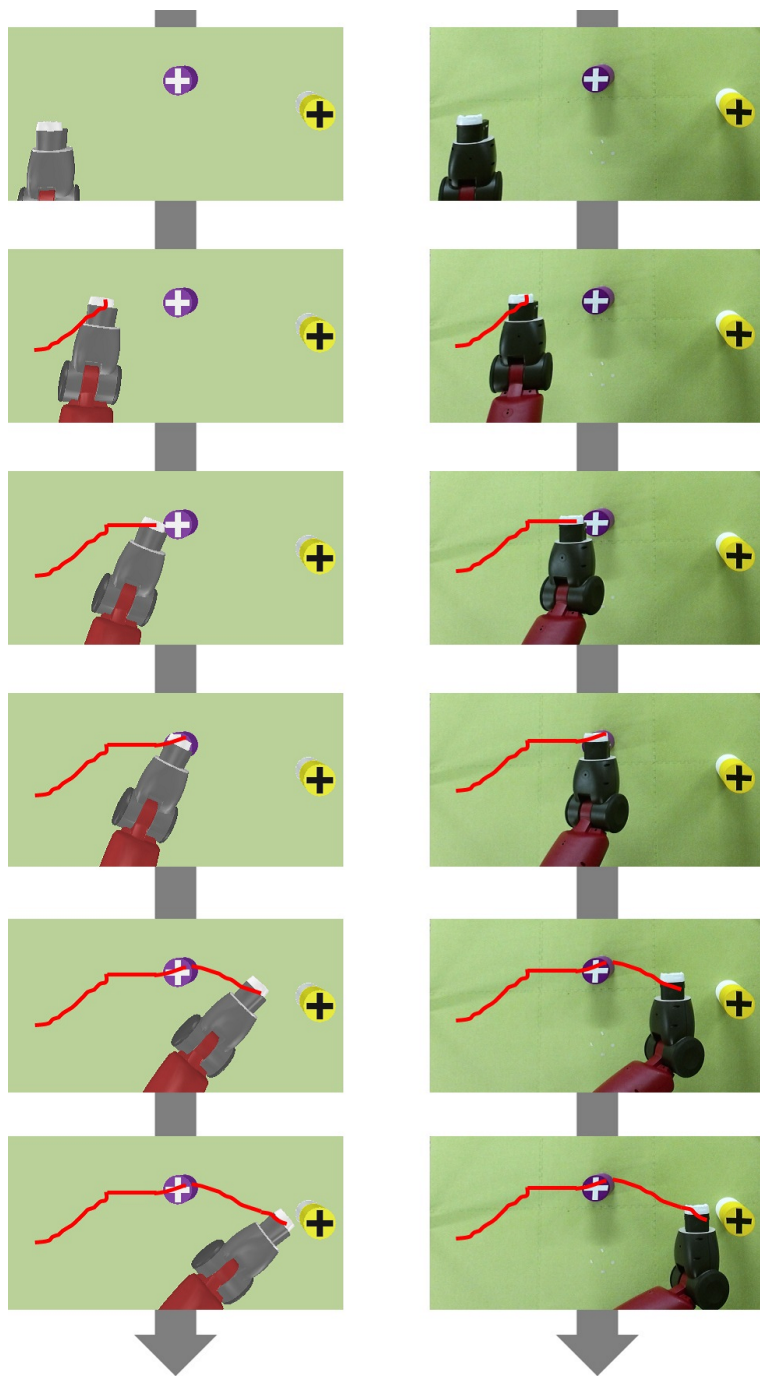


図 28 $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{real}}$ の対応の例 (上)

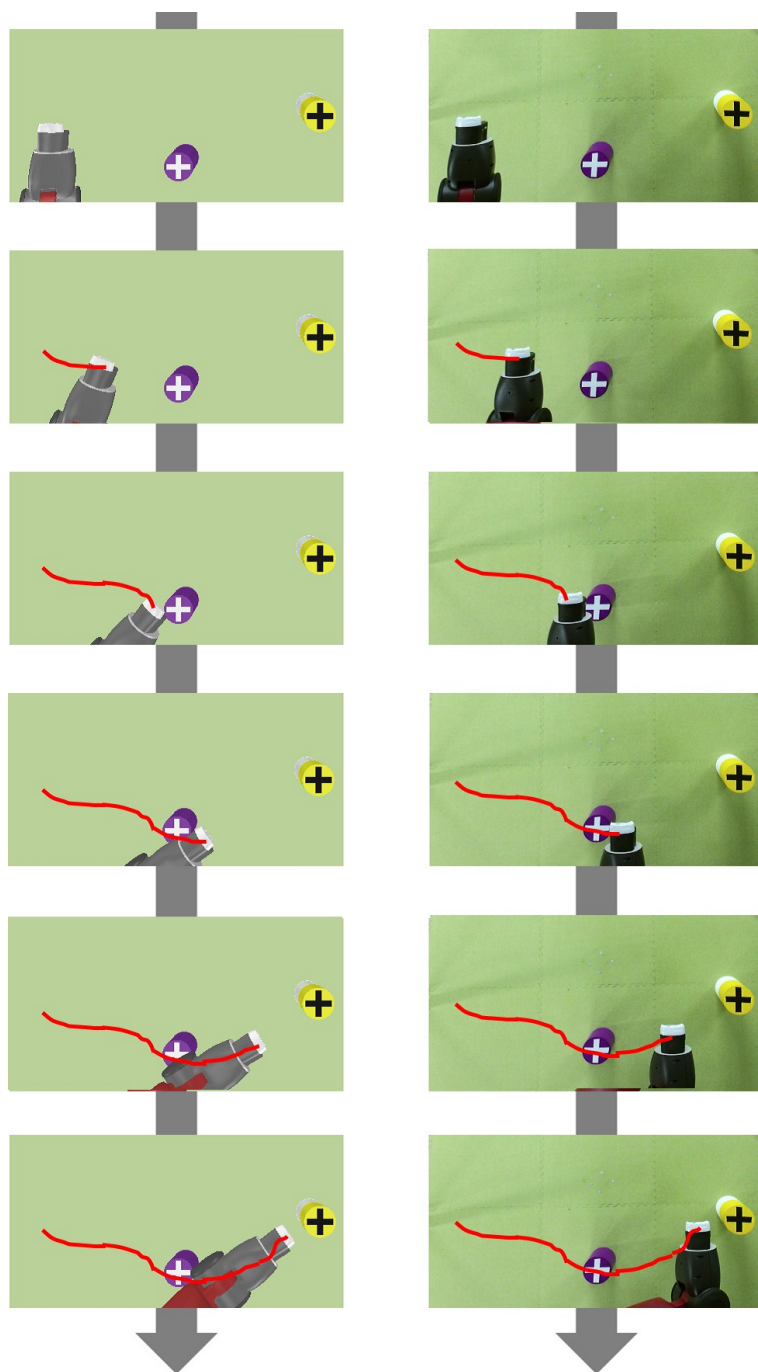


図 29 $\mathcal{S}_{\text{simu}}$ と $\mathcal{S}_{\text{real}}$ の対応の例 (下)

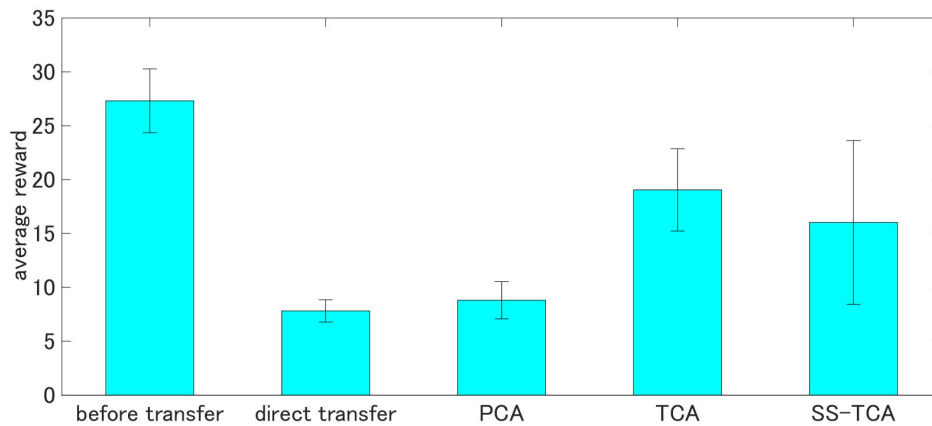


図 30 実験結果（実環境）

5.2.3 実験結果

実験結果を図 30 に示す．エラーバーは標準偏差を表す．また，各手法におけるタスク実行例を図 31，32 に示す．赤線は手先の軌道である．図 30 の直接転移の結果から，実環境に方策を直接転移したことで，方策の性能が劣化したことが分かる．直接転移を行った場合，図??の直接転移の例のように，全ての方策において，どの状況の画像を入力しても手先を下に運ぶ行動が得らるようになった．そのため，直接転移の平均報酬値の標準偏差は非常に小さい．PCA を利用して方策転移を行った場合は，直接転移の場合と同様に手先を下に運ぶ行動が生成されることが多かった．直接転移と比較すると，手先を少しずつ下に運ぶような行動が生成されたため，少しだけ平均報酬値が高いものの，方策転移が行われたとはいえない．TCA や SS-TCA を用いて提案法による方策転移を行った場合，いくらかの性能劣化が見られるものの，ある程度方策転移が行えている．特に TCA による方策転移を行った場合，図 31 の TCA の例のように，目標地点まで手先を運ぶことが出来る場合もあるものの，通過地点上を通過することが出来ないものが多く見られた．また，図 19 と比べて平均報酬値の平均が減少している．これは，図 28 から分かるように，シミュレーションと実環境で Baxter に同じ行動を実行させた場合に，手先位置は同様であるが姿勢が異なるなど，図 13 より大幅

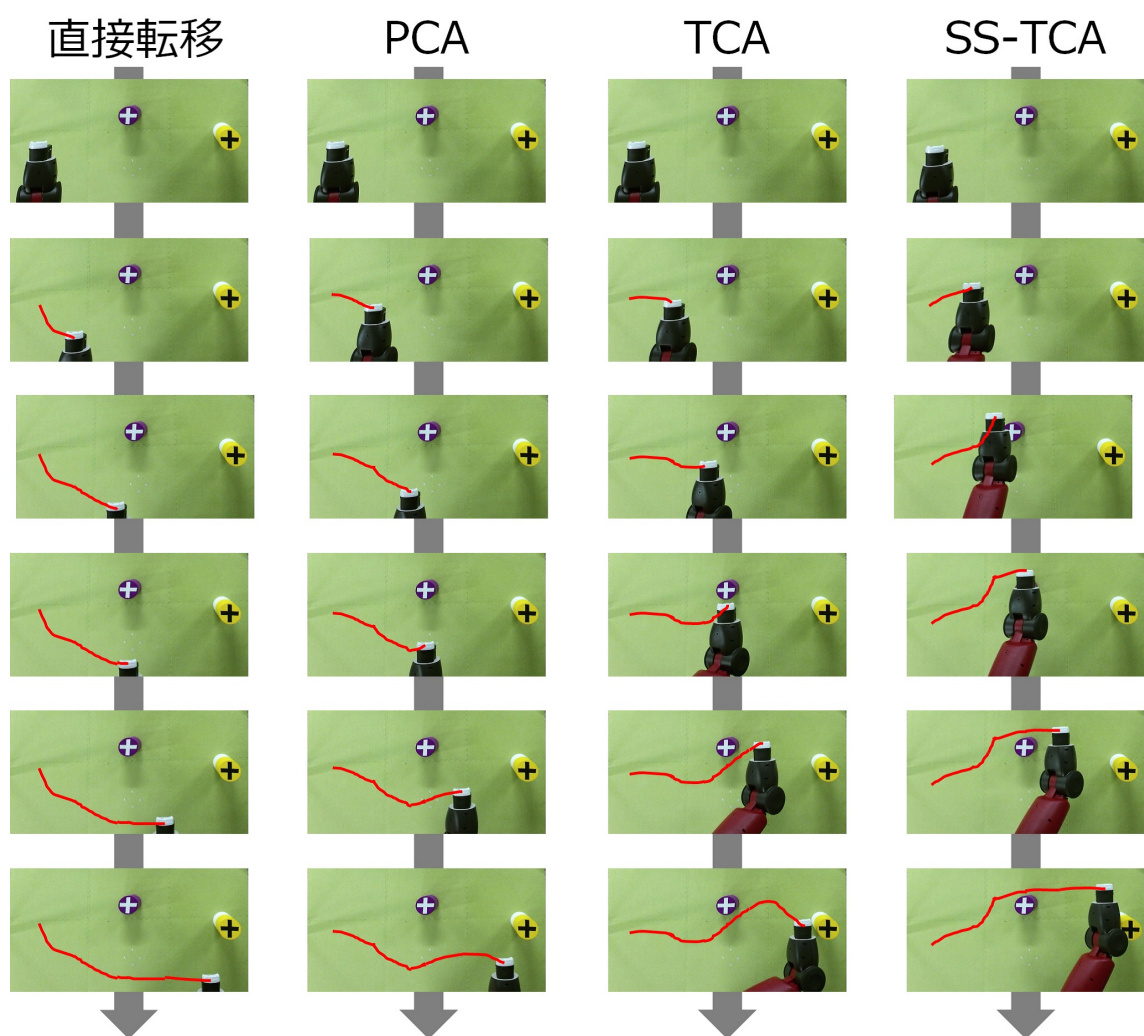


図 31 タスク実行例（上）

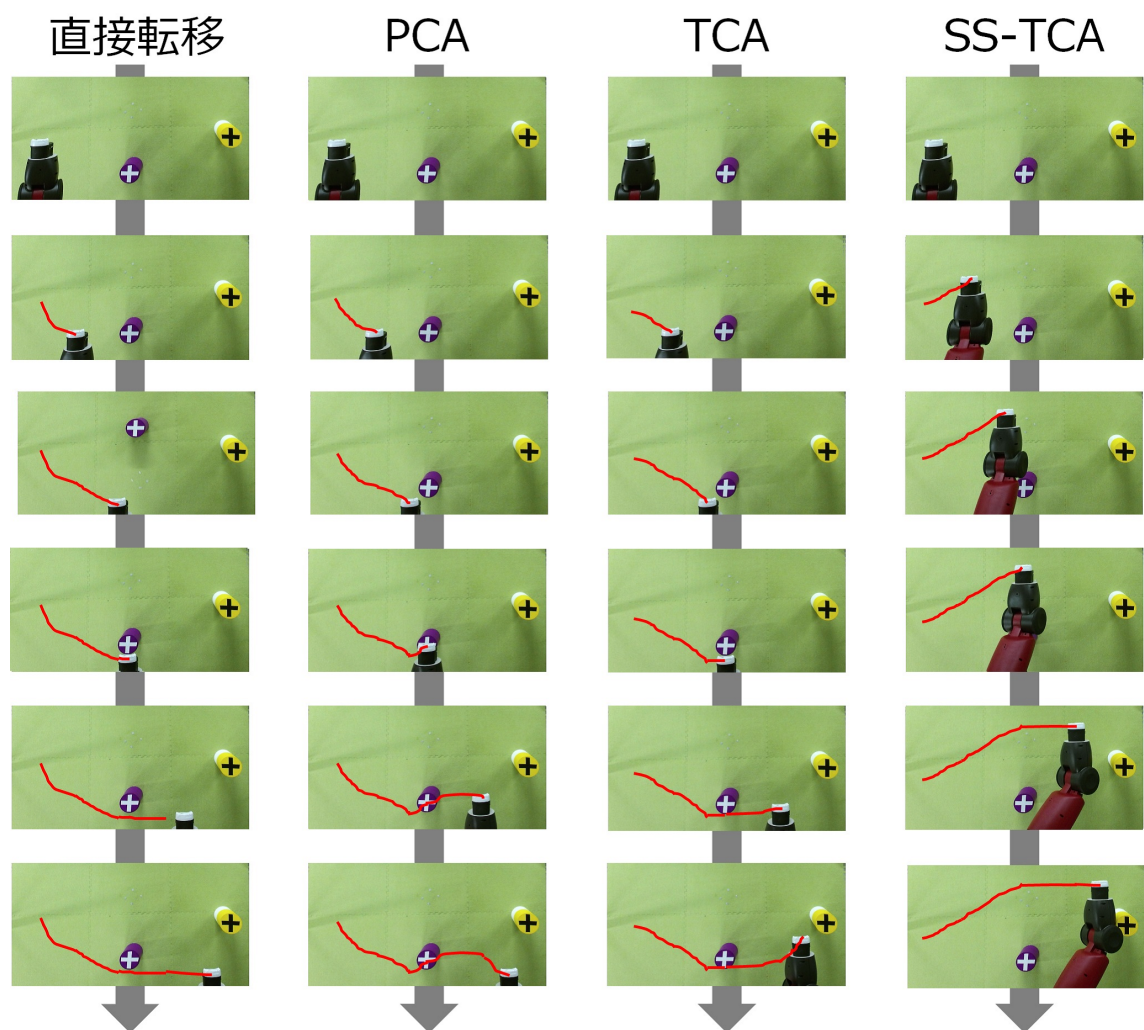


図 32 タスク実行例（下）

に見た目の異なる画像が得られたため、 S_{simu} と S_{real} の差異が大きくなり、シミュレーションと実環境の共通点を取りだした特徴量 ϕ を抽出できなかったためであると考えらる。その結果、見かけ上の誤差を含むシミュレーション環境に方策転移を行った場合に比べて、平均報酬値の平均が小さく、標準偏差が大きくなっている。SS-TCA に関しては、シミュレーションと比べて、平均値は大差ないが、標準偏差が大きくなっている。これは、3つの方策のうち1つにおいて、図 31, 32 の SS-TCA の例のようにどの状況の画像を入力しても、手先を上に乗ぶ行動が生成されるようになり、タスクを十分に実行できなくなったためである。

また、前節と同様に $\phi(s_{simu})$ と $\phi(s_{real})$ の類似度を計測した。以下がその結果である。

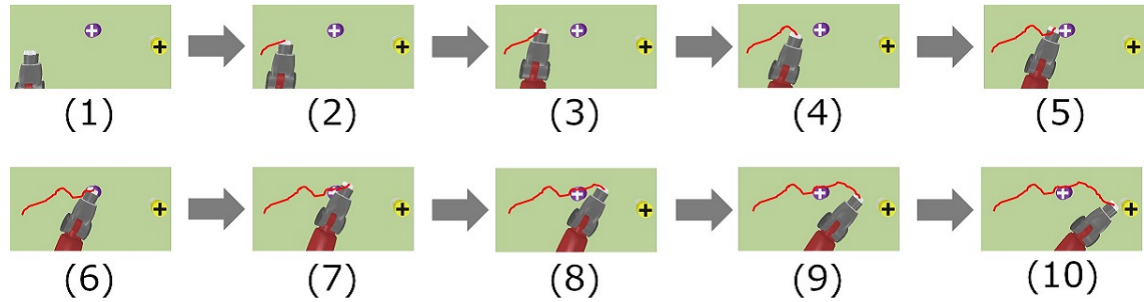


図 33 類似度計測用画像（シミュレーション）

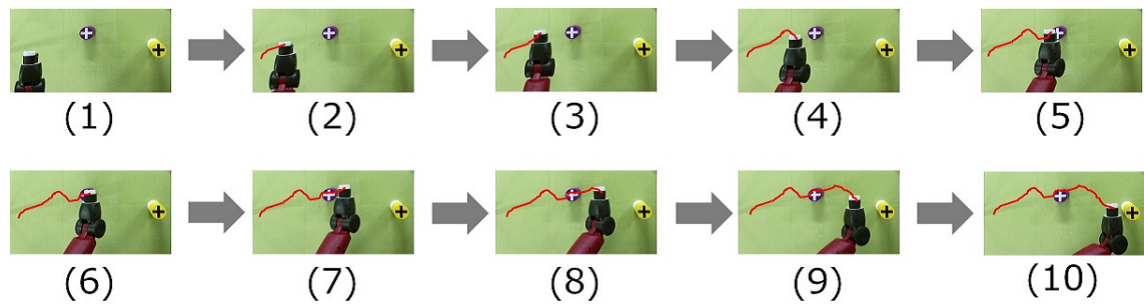


図 34 類似度計測用画像（実環境）

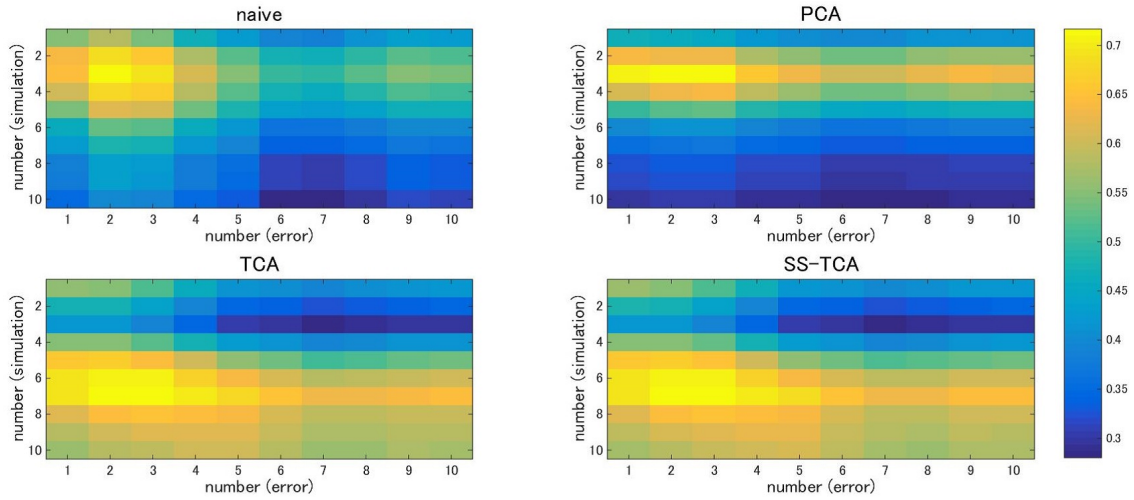


図 35 各手法による特徴量の類似度の比較

図の対応は前節と同様である．図 35 の naive を見ると，シミュレーションと実環境の類似度が非常に低く，差異が大きい．そのため，図 30 の直接転移の結果のように方策転移が行われなかった．図 35 の PCA を見ると，naive と比べて黄色の領域が少し増加している．そのため特徴量間の共通性が少し増加している．その結果，図 30 においても直接転移の例より PCA を用いたほうが平均報酬値が少しだけ高い．

TCA, SS-TCA に関しては，図 35 の TCA と SS-TCA から類似度が前節の実験より低いにもかかわらず方策転移がある程度行えている．このことから，方策転移の際には $\phi(s_{simu})$ と $\phi(s_{real})$ の類似度も重要ではあるが， $\phi(s_{simu})$ と $\phi(s_{real})$ が元のデータの分散を保持していることも重要であるといえる．また，報酬値と特徴量の相関を保存する SS-TCA を用いた提案法より，分散を保存する TCA を用いた提案法でのほうが平均報酬値が高い．そのため，前節の結果と同様にタスクの実行には報酬値以外にも重要な何かが存在し，それが元データの分散を保持することで保たれると考えられる．

以上から，シミュレーションで学習した方策を実環境に転移する場合，提案法に一定の有効性が確認できた．特に TCA を用いた提案法が有効であることが確認できた．前節と同様の結果が得られたことから，特徴抽出の基準として，報酬

値との相関より，データの分散を保存するように特徴抽出を行う方が方策転移には有効であることが分かった．

6. 結論

6.1 まとめ

本研究では、シミュレーションにおいて得られた報酬値を利用して、シミュレーションと実環境の見かけ上の誤差と方策の性能低下に頑健な特徴量を抽出し、抽出された特徴量を方策の入力として、シミュレーションにおいて再度強化学習を行うことで、実環境での学習を行わずに実環境に転移可能な方策を学習する手法を提案した。シミュレーションと実環境には、見かけ上の誤差が存在するため、SPM を用いて画像から汎用特徴量を抽出し、転移成分分析を用いることで、両環境に共通する特徴量を抽出した。抽出した特徴量を用いて、シミュレーション環境で再度強化学習を行うことで、実環境に転移可能な方策を生成した。

実験では2つの提案手法の有効性の検証を行った。初めに、シミュレーションで学習した方策を、見かけ上の誤差を伴うシミュレーション環境への転移を行った。結果から、見かけ上の誤差が比較的少ない環境に方策を転移する場合、TCA を用いた提案手法が有効であることが確認できた。SS-TCA を用いた提案手法では、方策転移は十分には行われなかったが、一定の有効性があることが確認できた。次いで、実環境に方策転移を行い、見かけ上の誤差が比較的多い環境であっても、TCA を用いた提案手法がある程度有効であることが確認できた。また、SS-TCA を用いた提案法では方策転移は十分には行われなかったが、一定の有効性があることが確認できたことから、方策転移に重要な報酬値以外の情報が存在する可能性が示唆された。

6.2 今後の課題

1つ目の課題として、第5章で述べた、見かけ上の誤差の影響で元環境で得られた特徴量が同様に検出されない場合は、SS-TCA を用いた提案法は有効ではないという仮説を検証するため、経路地点や目標地点のオブジェクトの模様を複雑化し、検出される特徴量を増加させてシミュレーションを行った。各オブジェクトから検出される特徴量が増加することで、見かけ上の誤差が加わった場合に検

出されなくなる特徴量が相対的に減少し、SS-TCA を用いた提案法の有効性が確認できると考えたためである。しかしながら、検出される特徴量を増加させても、SS-TCA を用いた提案法の有効性は確認できなかった。そこで、見かけ上の誤差の程度を軽減させた環境に対して方策転移を行った。見かけ上の誤差の軽減の程度は、学習した方策を直接転移した際に平均報酬値がどれだけ減少するか、を指標に設定した。結果として、直接転移の際の平均報酬値が約 15 になるまで見かけ上の誤差の程度を軽減した環境では、SS-TCA の有効性が確認出来た。以上から、SS-TCA を用いた提案法は見かけ上の誤差が大きすぎる場合は適応できないことが分かった。今後は、SS-TCA を用いた提案法が TCA を用いたものより有効となる見かけ上の誤差の程度を、直接転移の際の平均報酬値の低下の度合いによる確認等が更なる課題として挙げられる。

2 つ目の課題として、第 5 章で述べた、分散を保存することで保たれる、方策転移に重要な報酬値以外の情報がするという仮説の検証である。本研究では、タスクの可否を反映する報酬値と特徴量の相関を保つ特徴量を選択するだけでは方策転移を十分には行えない可能性が示唆された。この仮説が正しい場合、必要な情報を過不足なく保存するよう特徴量を抽出することで、提案法の性能をさらに向上させることが出来ると考えられる。そのため、本仮説の検証と、方策転移を行う際に重要な情報の特定を行いたい。

3 つ目の課題として、シミュレーションと実環境の Baxter の逆運動学の仕様を一致させることが挙げられる。図 28 からわかるように、シミュレーションで生成した行動を実環境において再現した場合、逆運動学の仕様が異なるため、手先の位置は同じになっているが、腕の姿勢が異なっている。その結果、図??に示すように、シミュレーションと実環境の特徴量の類似度が大幅に低下し共通する特徴量を抽出できておらず、方策転移の精度が劣化してしまった。実環境に方策転移を行う際の提案手法の有効性を正しく検証するには、逆運動学の仕様を一致させ、 $g_{simu}(s)$ と $g_{real}(s)$ の差異を出来る限り少なくして実験を行う必要がある。

4 つ目の課題として、タスクや環境の複雑化が挙げられる。今回はシンプルな環境下で、リーチングというシンプルなタスクを実行したが、複数のオブジェクトの存在する環境で特定のオブジェクトの把持など、環境やタスクを複雑するこ

とで，提案手法の有効性を検証したい．

謝辞

本研究を進めるにあたり，知能システム制御研究室の杉本謙二教授には研究のご指導に留まらず，研究者，技術者としての姿や考え方など，様々な面でご指導，ご鞭撻いただき，心より御礼申し上げます。

また，お忙しい中，副指導教員を引き受けていただき，ゼミナール発表で大変貴重なご意見を頂きましたロボティクス研究室の小笠原司教授に深く御礼申し上げます。

松原崇充准教授には，本研究の基礎となる知識，研究の進め方，論文の書き方や発表の仕方など，多方面のアドバイスや熱心なご指導を賜りました。改めて厚く御礼申し上げます。

定例研究会においてご指摘や発表に関する様々なアドバイスをいただきました小蔵正輝助教に深く感謝いたします。

研究に関するご指摘や，研究活動を行う上での相談・サポートをいただきました小林泰介助教に深く感謝します。

学会参加に関する事務処理や，その他学生生活を送るうえで様々なサポートをしていただいた秘書の林英子さんに深く感謝いたします。

また，学生生活を支えて頂いた両親と，研究面において様々なアドバイスをしてくれた知能システム制御研究室の皆様心より感謝致します。

参考文献

- [1] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, Vol. 17, No. 39, pp. 1–40, 2016.
- [2] 鶴峯義久, 崔允端, 内部英治, 松原崇充. 生画像を入力とするサンプル効率の良い深層強化学習と双腕ロボットによる布操作への応用. 日本ロボット学会学術講演会, 2017.
- [3] Sergey Levine, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with large-scale data collection. In *International Symposium on Experimental Robotics*, pp. 173–184, 2016.
- [4] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3406–3413, 2016.
- [5] Stephen James and Edward Johns. 3d simulation for robot arm control with deep q-learning. *arXiv preprint arXiv:1609.03759*, 2016.
- [6] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1912–1920, 2015.
- [7] Edward Johns, Stefan Leutenegger, and Andrew J. Davison. Deep learning a grasp function for grasping under gripper pose uncertainty. *arXiv preprint arXiv:1608.0223*, 2016.
- [8] Ignasi Clavera, David Held, and Pieter Abbeel. Policy transfer via modularity and reward guiding. In *International Conference on Intelligent Robots and Systems (IROS)*, 2017.

- [9] Andrei A. Rusu, Matej Vecerik, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. Sim-to-real robot learning from pixels with progressive nets. *arXiv preprint arXiv:1610.04286*, Vol. abs/1610.04286, , 2016.
- [10] Joshua 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. *arXiv preprint arXiv:1703.06907*, 2017.
- [11] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, Vol. 22, No. 2, pp. 199–210, 2011.
- [12] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, Vol. 60, No. 2, pp. 91–110, 2004.
- [13] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE computer society conference on Computer vision and pattern recognition(CVPR)*, Vol. 2, pp. 2169–2178, 2006.
- [14] Hirotaka Hachiya, Jan Peters, and Masashi Sugiyama. Reward-weighted regression with sample reuse for direct policy search in reinforcement learning. *Neural Computation*, Vol. 23, No. 11, pp. 2798–2832, 2011.
- [15] Kristen Grauman and Trevor Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, Vol. 2, pp. 1458–1465, 2005.
- [16] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, Vol. 7, No. 1, pp. 11–32, 1991.

- [17] Arthur Gretton, Karsten M Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J Smola. A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pp. 513–520, 2007.
- [18] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Scholkopf. Measuring statistical dependence with hilbert-schmidt norms. In *International conference on algorithmic learning theory*, pp. 63–77, 2005.
- [19] Alex Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A hilbert space embedding for distributions. In *International Conference on Algorithmic Learning Theory*, pp. 13–31, 2007.
- [20] Ingo Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of machine learning research*, Vol. 2, No. Nov, pp. 67–93, 2001.
- [21] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, Vol. 10, No. 5, pp. 1299–1319, 1998.
- [22] Hirotaka Hachiya and Masashi Sugiyama. Feature selection for reinforcement learning: Evaluating implicit state-reward dependency via conditional mutual information. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 474–489, 2010.
- [23] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, Vol. 3, 2009.
- [24] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, Vol. 2, No. 1-3, pp. 37–52, 1987.