

穴位置不確実 Peg-in-hole のための疑似力覚を組み込む深層模倣学習

○丹羽 靖治[†], 相澤 宏旭[‡], 加藤 邦人[†]

[†]: 岐阜大学工学部

[‡]: 広島大学先進理工系科学研究科

niwa@cv.info.gifu-u.ac.jp

概要：穴の位置不確実性が大きい Peg-in-hole では、ペグの挿入のために、時間をかけて正確に穴を探索する必要があった。本研究では、提案する疑似力覚モジュール PHM と特徴抽出器 CNN⁴ を組み込む深層学習モデルを人間の実演から訓練することで、ロボットアームに人間のような「探りながらの挿入」を模倣させ、正確な穴の位置なしで Peg-in-hole を達成した。
＜キーワード＞ Peg-in-hole, 模倣学習, 疑似力覚

1. はじめに

挿入物をわずかな余裕を持つ穴に挿入する組み立て作業は Peg-in-hole と呼ばれる。クリアランスを大きく上回る位置不確実性のある穴に対しペグを挿入するために、これまで様々な研究が行われてきた。既存手法では、穴の探索とペグの挿入の2つに問題を切り分け解決を試みている。しかし、挿入のためには、正確に穴の位置を特定する必要があり、その作業に時間を要する点が課題となっている。本研究ではこの課題を解決するために、模倣学習で人間の実演からロボットアームの方策を学習することで、人間のような「探りながらの挿入」を模倣させ、正確な穴の位置なしで Peg-in-hole を行う手法を提案する。

ペグを挿入するとき、指や腕で感じる力覚が挿入のガイドとして大きな役割を果たすが、模倣学習で力覚を考慮した判断をロボットに組み込むには、Expert へ力覚をフィードバックする高価で複雑な操作機器が必要となる。本研究では、この問題を解決するために、ヒューマンインターフェースの分野で研究されている「疑似力覚」に着目した。疑似力覚とは、ユーザー操作と視覚情報との乖離から生起する錯覚的な力覚を指し、特別な操作機器なしでユーザーに力覚を提示できる。

提案する疑似力覚モジュール (Pseudo-haptics Module, PHM) は、時刻 $t-1$ でモデルが出力した過去の操作値 $\mathbf{a}^{t-1} \in \mathbb{R}^3$ と、モデルが予測したペグ中心の並進速度 $\mathbf{e}^t \in \mathbb{R}^3$ との差分をとり、乖離量と乖離方向を理解するための疑似力覚 $\mathbf{h}^t \in \mathbb{R}^3$ を得る。その後、LSTM に \mathbf{a}^{t-1} , \mathbf{e}^t , \mathbf{h}^t から得られた特徴量を蓄積することで、ネットワークに過去の行動、結果、乖離を考慮した次の行動 \mathbf{a}^t を出力させる。図1に \mathbf{a}^{t-1} , \mathbf{e}^t , \mathbf{h}^t の関係を示す。PHM の疑似力覚の計算には、ペグ中心の並

進速度を正確に予測する必要があるため、本研究では2フレームの左右画像から、空間的、時間的差分を考慮した特徴抽出を行う CNN⁴ (CNN to the Fourth Power) を提案し、PHM のバックボーンとして使用する。

実験では、構築した操作システムで人間の実演を収集し、モデルの訓練に使用した。シミュレータを用いた実験の結果、PHM と CNN⁴ を組み込む提案ネットワークの Peg-in-hole 成功率は、制限時間を 60[秒]としたとき、97.2[%] (972/1000 [回]) となり、探索から挿入までの平均所要時間は 10.0[秒] となった。

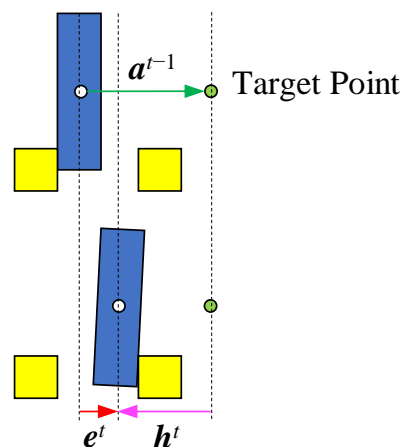


図1: 横から見たペグと穴の断面図。青色の長方形はペグ、黄色の正方形は穴の側面を意味する。時刻 $t-1$ では、ペグの中心座標に操作値 \mathbf{a}^{t-1} を加算した座標にロボットアームの目標点を置く。時刻 $t-1$ から t までの間にロボットアームは、ペグ中心を目標点に重ねようと動く。しかし、ペグが穴に衝突するため、ペグ中心と目標点は重ならない。時刻 t では、ペグ中心の並進速度 \mathbf{e}^t をモデルが予測し、過去の操作値 \mathbf{a}^{t-1} との差分をとり、疑似力覚 \mathbf{h}^t とする。

2. 関連研究

2.1. Peg-in-hole

穴の位置不確実性の大きい Peg-in-hole は一般的に穴の探索とペグの挿入の 2 段階が必要とされ、先行研究の多くでも別の問題として扱われてきた。

探索段階では、位置の不確実性を解消するために、センサを使用したペグと穴の位置合わせをする。使用するセンサによって、視覚センサ系、力覚センサ系、統合系の 3 つのカテゴリに大別される。視覚センサ系では、画像から穴を検出し、挿入を行うため、検出誤差が挿入に失敗につながる。文献[1]は、画像からの穴検出と、3 次元面の再構成を組み合わせることで、不確実な位置と傾きを持つ穴を特定した。穴の検出は非常に高速だが、ロボットアーム先端のカメラで、対象物を様々な角度から撮影する必要がある、カメラの移動に時間がかかる点が課題となる。文献[2]は、力覚センサを用いたブラインドな探索手法を複数提案した。その 1 つ Spiral Path Search は、渦状軌跡で全探索を行い、力覚センサ値の変化をとらえることで穴の位置を特定する。ブラインドサーチの欠点は、穴から遠い位置で探索を開始した場合、特定までに長い時間を要することである。文献[3]は視覚センサと力覚センサを統合した探索手法を提案した。ロボットアーム先端に取りつけられたカメラ画像から、大まかな位置関係を識別することで反復的にペグを穴の近傍へ遷移させ、最後に Spiral Path Search で穴の特定を行った。

挿入段階では、位置合わせされたペグを穴へ挿入する。一般的に使用される手法は、インピーダンス制御[4]である。インピーダンス制御は、RCC ハンド[5]のような専用機構なしで、ロボットアームに人間の腕のような「しなやかさ」を与える制御法である。センサで得られた位置や外力情報から、目標点に柔軟に向かうための関節駆動量を得る。インピーダンス制御のようなルールベースの挿入手法の他に、学習ベースで挿入を自動化した研究もある。文献[6]は深層強化学習で挿入を自動化し、文献[7,8]は模倣学習を使用した挿入手法を提案した。しかし、これら先行研究は、穴の位置が既知な問題設定で挿入を行っているため、穴の位置不確実性が大きい Peg-in-hole では穴の探索が必須となる。

2.2. 模倣学習

模倣学習(Imitation Learning)[9]は、熟練操縦者の実演(Demonstration)からロボットや自動車が自律的に行動するための方策を学習する手法である。先行研究では、不確実な要素を持つ環境に対し、人間のような柔軟な判断を機械に組み込めることが示されてきた[10,11,12,13,14]。模倣学習の分野では、熟練操縦者を Expert、ロボットや自動車などの学習者を Agent と呼ぶ。実演は Agent の状態 s と Expert が与える正解行動 p の組 (s, p) として表される。学習パラメータ θ を持つ方策 $\pi(s; \theta)$ は、式(1)のように状態 s を入力し、行動 u を出力する関数である。正解行動 p を教師信号とした教師あり学習で θ の最適化を行う。方策に深層学習モデルを適用したものを深層模倣学習(Deep Imitation Learning)と呼び、近年では、ニューラルネットワークの発展に伴い、カメラから得られた生画像を状態として入力し、行動に直接マッピングする方策を設計することもできる。

$$u = \pi(s; \theta) \quad (1)$$

模倣学習をロボットアームに適用するには、リアルタイムな操作システムを設計し、実演を収集する必要がある。画像を方策の入力とする模倣学習では、VR 機器を使用した遠隔操作で実演を収集するのが一般的である[11,12,13]。文献[11]は VR 機器を使用してロボットを遠隔操作し、高品質な実演を収集することで、少量データから様々なマニピュレーションタスクで高い成功率を達成した。Expert への立体的な視覚提示は、深度カメラから得られる 3 次元点群を仮想空間上に配置することで行った。ロボットアームの目標点の操作には、VR コントローラを用いた。文献[12,13]は、2 眼カメラから得られる RGB 画像をそのまま VR ゴーグルの左右の画面に独立して提示する方法を採用している。しかし、VR 機器を使用した遠隔操作手法は視覚フィードバックのみであり、実際の力覚を考慮した実演を収集できない欠点がある。

模倣学習で実際の力覚を考慮した判断を Agent に組み込むには、力覚を操縦者へフィードバックする操作機器が必要になる。近年、力覚と位置を同時に制御できるバイラテラル制御でロボットアームを遠隔操作し、実演を収集した模倣学習の研究もある[15,16,17]。しかし、力覚フィードバックを搭載した操作機器は高価で複雑であり、市販のものは可動域も狭い。

2.3. 疑似力覚

人間の体制感覚は、体表で感じる皮膚感覚(触覚)と筋肉や関節で感じる深部感覚(力覚)の2つに大別される。これらの感覚を特別な操作機器なしで、錯覚的にフィードバックする方法がヒューマンインターフェースの分野で研究されている[18,19,20,21]。この錯覚的な触力覚を疑似触力覚(Pseudo-haptics)と呼ぶ。疑似触力覚は、ユーザー操作に応じて視覚情報を歪ませることで生起させることができる。特に、力覚に分類される「重み」をユーザーに感じさせるためには、変位による操作と、変位による視覚提示を選択し、その間に乖離を設けることで疑似力覚を生じさせる[18,19,20]。例として、1次元のマウスポインタ操作を考える。ユーザーのマウス速度を $\mathbf{o} \in \mathbb{R}^1$ 、ディスプレイ上のポインタの速度を $\mathbf{d} \in \mathbb{R}^1$ とし、これらの比を式(2)のように計算する。式(2)の比は C/D 比(Control / Display Ratio)と呼ばれる。画面に表示するポインタの速さ $|\mathbf{d}|$ を小さくし、C/D 比を大きくすると、ある時点でユーザーは操作と逆方向に力を感じる。逆に $|\mathbf{d}|$ を大きくして C/D 比を小さくしていくと、ユーザーはある時点で操作方向に引っ張られる力を感じる[21]。

$$\frac{|\mathbf{o}|}{|\mathbf{d}|} \quad (2)$$

3. 提案手法

本研究では、節3.1に示す提案ネットワークをロボットアームの方策とする。節3.2に示す操作システムで実演を収集し、方策の訓練に使用する。実演から訓練したロボットアームは、人間のような「探りながらの挿入」を模倣し、正確な穴の位置なしで、探索と挿入をまとめて End-to-end に達成することができる。

3.1. ネットワーク

図2に提案ネットワークの構造を示す。ネットワークは、特徴抽出器(CNN⁴)、疑似力覚モジュール(PHM)、行動出力器(Action Network)の3つからなる。

特徴抽出器 CNN⁴ について説明する。CNN⁴ は CNN²[22]を基本としている。CNN² は人間の視差による両眼立体視と、焦点ずれによる単眼立体視を明示的に組み込んだモデルである。図3のように、左右の特徴マップの差分を連結した特徴マップを作り、Concentric Multi-Scale Pooling (CM pool)で解像度を落とした後、畳み込みを行う。提案する

CNN⁴ は2段の CNN² からなる。1段目の CNN² には、時系列左右画像 ($\mathbf{L}^{t-1}, \mathbf{R}^{t-1}$), ($\mathbf{L}^t \in \mathbb{R}^{256 \times 256 \times 3}, \mathbf{R}^t \in \mathbb{R}^{256 \times 256 \times 3}$) を入力し、特徴マップ $\mathbf{F}^{t-1}, \mathbf{F}^t \in \mathbb{R}^{32 \times 32 \times 256}$ を得る。2段目の CNN² には、 $\mathbf{F}^{t-1}, \mathbf{F}^t$ を入力し、時間的な差分をとりながら畳み込む。出力された特徴マップに Global Average Pooling (GAP)を適用し、 $\mathbf{f}^t \in \mathbb{R}^{4096}$ を得る。ここまでの CNN⁴ である。 \mathbf{f}^t をユニット数 512 の全結合層へ通し、ペグ中心の並進速度の予測に有用な空間的、時間的差分を考慮した特徴量 $\mathbf{g}^t \in \mathbb{R}^{512}$ を得る。

PHM について説明する。はじめに \mathbf{g}^t からペグ中心の並進速度を予測し、補助出力 \mathbf{e}^t を得る。次に、式(3)に従って、操作と結果の乖離を示す疑似力覚 \mathbf{h}^t を求める。 $\mathbf{a}^{t-1}, \mathbf{e}^t, \mathbf{h}^t$ を縦方向に連結し、特徴マップ $\mathbf{G}^t \in \mathbb{R}^{3 \times 3 \times 1}$ を得る。このとき、 $\mathbf{e}^t, \mathbf{h}^t$ の計算グラフを切ることで、最終層から補助出力層へ、ペグ中心の並進速度の予測に関係のない誤差が逆伝搬することを防ぐ。 x, y, z 軸別に $\mathbf{a}^{t-1}, \mathbf{e}^t, \mathbf{h}^t$ の関係性を認識させるため、 \mathbf{G}^t をカーネルサイズ(1,3)の縦方向フィルタで畳み込み、 $\mathbf{I}^t \in \mathbb{R}^{3 \times 1 \times 32}$ を得る。 \mathbf{I}^t を1次元ベクトル $\mathbf{i}^t \in \mathbb{R}^{96}$ に直し、LSTM へ入力する。最後に LSTM から、時系列的な行動、結果、乖離を考慮した特徴量 $\mathbf{j}^t \in \mathbb{R}^{128}$ が出力される。

行動出力器は、全結合層2層からなるネットワークであり、 \mathbf{g}^t と \mathbf{j}^t を連結した特徴量を入力し、次の操作値 \mathbf{a}^t を出力する。

$$\mathbf{h}^t = \mathbf{e}^t - \mathbf{a}^{t-1} \quad (3)$$

PHM は実行テスト時と学習時で挙動が異なる。図2右に CNN⁴ を省略した学習時の PHM と Action Network を示す。安定的に乖離と出力の関係性を学習させるために、 \mathbf{h}^t を計算するとき、50%の確率で \mathbf{e}^t を補助出力の教師信号 $\mathbf{r}^t \in \mathbb{R}^3$ に置き換える。過去の操作値 \mathbf{a}^{t-1} は訓練時に得られないため、Expert の過去の操作値 $\mathbf{c}^{t-1} \in \mathbb{R}^3$ に置き換える。LSTM 部は、系列長を固定して学習する。データセットから、 q ステップ時刻を遡った $\mathbf{r}^{t-q}, \mathbf{c}^{t-q-1}$ を得る。 $\mathbf{r}^{t-q}, \mathbf{c}^{t-q-1}, (\mathbf{r}^{t-q} - \mathbf{c}^{t-q-1})$ を縦方向に連結し、 $\mathbf{H}^{t-q} \in \mathbb{R}^{3 \times 3 \times 1}$ を作る。時刻を遡りながら \mathbf{H}^{t-q} を T 個用意し、学習時 \mathbf{G}^t の代わりとして入力する。補助出力の教師信号は \mathbf{r}^t 、最終出力の教師信号は操作値 \mathbf{c}^t であり、どちらも損失関数は RMSE を使用する。誤差逆伝搬は補助出力、最終出力の順に行い、2つの誤差の勾配を用いて重み更新を行う。

4. 実験

Expert の実演から学習したロボットアームが 2 フレームの左右画像(L^t, R^t), (L', R')と過去の出力 a^{t-1} の限られた情報から, 穴の探索とペグの挿入をまとめて行うことができるか, 所要時間はどれほどか確かめる実験を行う. 実験 A では, 実験 B の前段階として, PHM のバックボーンとなる特徴抽出器 CNN⁴ がペグの並進速度の予測に有用な特徴量を抽出できるか確かめる. 実験 B では, 提案ネットワークを収集した実演から訓練し, 実際に Peg-in-hole を行わせる. 各モデルに同一の 1000 パターンの未知の位置に設置された穴を与え, テストを行い, Peg-in-hole の成功率と, 所要時間で評価する. 節 4.1 で実験環境について詳細な説明を行い, 節 4.2 で収集したデータセットについて説明する. 節 4.3 で実験 A, 節 4.4 で実験 B の結果を示す.

4.1. 実験準備

実験のために Peg-in-hole のシミュレータを構築した. ロボットアームの模倣学習では, Expert と Agent 間の遠隔操作ループをリアルタイムかつ高速に行い, 実演を収集する必要がある. シミュレータでは, 画像レンダリングがボトルネックとなるため, 本環境では, 画像のレンダリングに特化した PyRep[23] ベースの CoppeliaSim[24]を使用する.

図 4 左のように, ロボットアーム, ペグ, 穴, 2 眼カメラを配置する. ペグはロボットアームの先端に剛体として固定される. 穴は原点から縦横方向 $[-10, 10]$ [mm] の範囲でランダムな位置に固定される. 2 眼カメラは, $(x, y, z) = (0, -90, 40)$ [mm]の位置に固定される. カメラの左右間隔は 10 [mm], 画角は 60 度に設定する. ロボットアームは, ペグの中心座標が目標点に向かうように動作する. ロボットアームの 6 つの関節は PI 制御で動作し, 多少の抵抗であれば I 制御の効果で乗り越えることができる. すべての関節は比例ゲイン $K_p = 0.1$, 積分ゲイン $K_i = 0.005$ に設定されている.

シミュレータでは物体が貫通しないように衝突面に対し, 衝突マージンを設ける必要がある. 本環境では, ペグと穴の衝突マージンは 0.1[mm]に設定されている. 衝突マージン同士の重なりは相殺されるため, 物理演算上のペグと穴のサイズ, クリアランスは表 1 のようになる. ここでのクリアランスは, ペグを穴の中心に置いたとき, ペグの周りにできる空間の大きさを指す. 制限時間内に, 図 5 に示すペグの先端座標と穴底の中心座標との距離が 3.0[mm]以下になったとき, Peg-in-hole が成功したとみなし, シミュレータを終了する.

表 1 ペグと穴のサイズとクリアランス

	外観サイズ [mm]	衝突マージンと マージン相殺を 考慮したサイズ [mm]
ペグ	9.6×9.6×50.0	9.8×9.8×50.2
穴	10.0×10.0×10.0	9.8×9.8×10.0
クリアランス	0.2	0.1

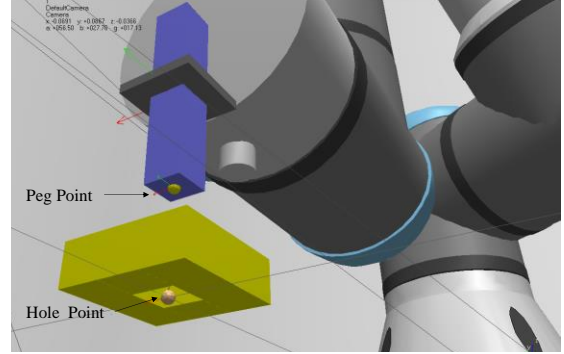


図 5: ペグの先端座標と穴の中心座標. ペグの先端座標(図中 Peg Point)と穴底の中心座標(図中 Hole Point)との距離で成功判定を行う.

4.2. データセット

Expert が操作システムを使用して, シミュレータ上のロボットアームを遠隔操作することで, 実演 $\{(L', R'), c'\}$ と, ペグ中心の並進速度の実測値 v を収集する. データ収集時にのみペグの初期位置を, $(x, y, z) = (0, 0, 50)$ [mm]の各軸に, $[-10, 10]$ [mm]の範囲を持つランダムオフセットを加えた座標とする. 初期位置をランダムにすることで, データ内に様々な状態を作り, 実行テスト時の頑健性を向上させる.

500 パターンの異なる位置を持つ穴に対し, 探索と挿入を行うことで, 合計 500 シーンのデータを収集する. その後, シーン単位でランダムに, 425 シーンを訓練データ, 37 シーンを検証データ, 残り 38 シーンを評価データに割り当てる. 図 6 にそれぞれのデータに割り当てられた穴の位置を示す. モデルの学習に訓練データを使用し, 重みパラメータの選択に検証データを用いる. 評価データは, モデルがペグの並進速度を予測する能力を評価するのに用いる. ロボットアームに Peg-in-hole を行わせる実験 B では, データセットに含まれない 1000 パターンの穴の位置を使用する. 表 2 に入力と教師信号の組数を示す.

表 2 入力と教師信号の組数

訓練データ数	25156
検証データ数	2492
評価データ数	2033

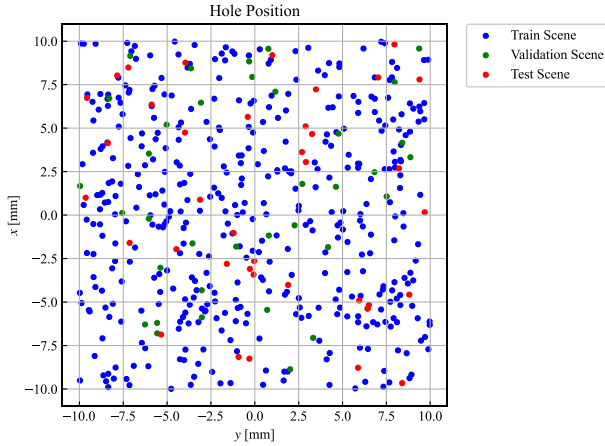


図 6: 訓練, 検証, 評価データの穴の位置. 青色の点が訓練データ, 緑色の点が検証データ, 赤色の点が評価データに含まれる穴の中心座標(x, y).

4.3. 実験 A: ペグ並進速度の予測精度比較

実験 A では, CNN^4 が 2 フレームの左右画像(\mathbf{L}^{t-1} , \mathbf{R}^{t-1}), ($\mathbf{L}^t, \mathbf{R}^t$)から, ペグの並進速度の予測に有用な特徴量を抽出できるか, 補助出力のみを学習することで確かめる. 図 7 に CNN^4 の比較対象として使用する 2 つの特徴抽出器 Vanilla と CNN^2 を示す.

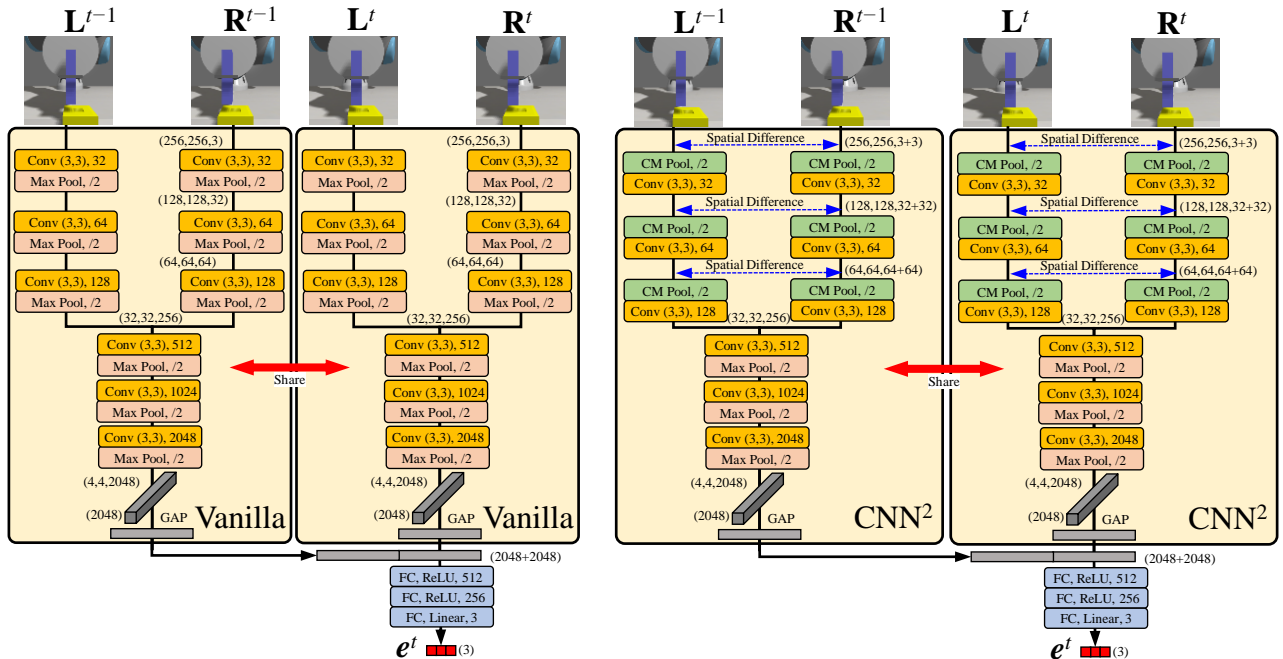


図 7: 比較する特徴抽出器. 左が Vanilla, 右が CNN^2 . 全結合層 3 層でペグ中心の並進速度を予測する.

Vanilla は 2 フレームの左右画像をシンプルに畳み込んだ後, GAP で 1 次元化した特徴量を出力する. CNN^2 も Vanilla と同様の流れで 2 フレームの左右画像から特徴抽出を行う. 各特徴抽出器で得た特徴量から, 3 層の全結合ネットワークでペグ中心の並進速度を予測する. 各モデルを, エポック数 50, バッチサイズ 64, 最適化手法 Momentum-SGD (モーメンタム 0.9), 損失関数 RMSE, 学習率 10^{-4} , 10^{-3} で学習を行い, 検証誤差が最も下がったエポックの学習パラメータを使用して, 評価データの RMSE を算出する.

表 3 に各ベストモデルの RMSE を示す. 表 3 から, モデル Vanilla の CNN を空間的な差分を考慮した CNN^2 に変更するのみでは多少の改善しか得られないことが分かり, 時間的な差分も考慮できる CNN^4 は他の 2 つと比較し, より小さな誤差でペグ中心の並進速度を予測ができることが判明した. ここから, 2 つの時刻で得られた特徴マップの差分をとりながら畳み込むことが, 物体の時間的な変化をとらえることに効果的であることが分かる. 実験 A の結果を受け, 実験 B では PHM バックボーンとして CNN^4 を選択する.

表 3 特徴抽出器別のペグ並進速度の予測誤差

特徴抽出器	RMSE
Vanilla	0.1222
CNN^2	0.0939
CNN^4	0.0379

4.4. 実験 B: Peg-in-hole

実験 B では、実際に Peg-in-hole をロボットアームに行わせ、モデル別の成功率を比較する。比較モデルとして特徴抽出器を Vanilla や CNN² に置き換えたモデルや PHM をなくしたモデルを選択する。特徴抽出器に Vanilla, CNN² を使用したモデルでは、入力画像を 1 フレームと 2 フレームに変更した実験も行う。エポック数 50, バッチサイズ 64, 最適化手法 Momentum-SGD (モーメンタム 0.9), 損失関数 RMSE, 学習率 10^{-4} , 10^{-3} で学習を行う。加えて、提案ネットワークでは、 H^q の個数 T を 5, 10, 20, LSTM のユニット数を 128, 256, 512 に変更して学習を行う。10 エポック間、検証誤差が改善しなければ学習を終了する。実行テストでは、最終出力の検証誤差が最も下がったエポックの重みを使用する。提案ネットワークのベストパラメータは、学習率 10^{-3} , $T = 20$, LSTM のユニット数 128 となった。

表 4 に各ベストモデルの制限時間別で見た Peg-in-hole 成功率を示す。表 5 に制限時間を 60[秒]としたときの所要時間の平均、標準偏差を示す。ただし、平均と標準偏差を計算時、失敗試行は所要時間を 60[秒]として計算する。

表 4 から、提案ネットワーク CNN⁴+PHM は、挿入成功率と挿入時間ともに 1 フレームの Vanilla から大幅な改善を示した。入力画像を 2 フレームにしたとき、Vanilla では成功率が大きく改善し、挿入時間も短くなるが、CNN² では、逆に成功率が低下したため、単純に 2 フレームにするだけでは、確実な改善につながらないことが判明した。2 フレームの CNN² と CNN⁴ を比較すると、成功率、所要時間の両方が大きく改善したことから、空間的差分を考慮した畳み込みが成功率改善に大きく寄与することが分かった。最後に CNN⁴ と CNN⁴+PHM を比較すると、CNN⁴+PHM は 15.0[秒]以内の成功率が 6.3[%](63[回]) 改善し、平均所要時間も 2 秒以上の改善した。この結果から、過去の操作 a^{t-1} , 結果 e^t , 乖離 h^t をモデルに与える PHM が、成功率と所要時間短縮に大きく寄与すると言える。

PHM を入れたモデルの明確な動きの違いとしては、穴の探索時に引っかかりによって動けなくなったとき、ペグを上にあげることができる点であった。失敗パターンは 2 つあり、1 つは穴がペグで完全に隠れていない状態で、あと少しの移動で入るとき、モデルはペグの引っかかりに関わらず、穴に向かう操作を続けてしまう。これは Expert の実演を改善することで解消する可能性がある。もう 1 つは、挿入時にペグと穴が引っかかり、脱出できないパターンであった。Expert はこの状態に陥ったとき、一貫性のない操作を行って脱出

できるが、モデルは統計的な行動を学習するため、Expert の行うランダムな動きまで模倣することはできず、失敗に至る。

表 4 制限時間別の Peg-in-hole 成功率[%]

制限時間[秒] \ モデル	15.0	30.0	45.0	60.0
Vanilla (1 frame)	70.5	78.4	86.0	89.5
Vanilla (2 frame)	74.5	83.5	89.2	92.3
CNN ² (1 frame)	74.1	82.4	90.0	92.9
CNN ² (2 frame)	75.2	84.9	88.7	91.3
CNN ⁴	77.1	89.0	94.2	95.6
CNN ⁴ +PHM	83.4	92.3	95.8	97.2

表 5 所要時間の分布

	平均[s]	標準偏差[s]
Vanilla (1 frame)	16.3	19.4
Vanilla (2 frame)	14.0	17.9
CNN ² (1 frame)	14.1	17.6
CNN ² (2 frame)	14.0	17.8
CNN ⁴	12.2	14.3
CNN ⁴ +PHM	10.0	12.4

5. まとめ

本稿では、穴の位置不確実性の大きい Peg-in-hole を自動化するために、模倣学習を使用して探索と挿入を一括して行う Peg-in-hole 手法を提案した。Expert の実演から学習したロボットアームは、穴の位置不確実性に柔軟に対応し、人間のように探索と挿入をまとめて処理できることを示した。また、PHM を導入することで、成功率が向上し、所要時間が短縮することを示した。しかし、本手法はシミュレータの実験で挿入成功率が 100%に至っていない。これはロボットアームの制御法の見直しを行うことで改善できる可能性がある。制御法を改善することで、挿入の難易度を下げることができ、実演の質も向上する。また、本稿では Peg-in-hole の既存手法との比較を行っていない。以上から、今後の課題は、制御法の改善と、同条件で既存手法と本手法を比較することである。

謝辞

本研究は、岐阜県航空宇宙産業生産技術人材育成・研究開発事業補助金の助成を受けた。航空宇宙生産技術開発センターの皆様へ心より御礼申し上げる。

参考文献

- [1] M. Nigro, M. Sileo, F. Pierr, K. Genovese, D. D. Bloisi, and F. Caccavale: Peg-in-Hole Using 3D Workpiece Reconstruction and CNN-based Hole Detection, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp.4235-4240, 2020.
- [2] S. R. Chhatpar and M. S. Branicky: Search strategies for peg-in-hole assemblies with position uncertainty, 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, Expanding the Societal Role of Robotics in the Next Millennium (Cat.No.01CH37180), pp.1465-1470, 2001.
- [3] J. C. Triyonoputro, W. Wan, and K. Harada: Quickly inserting pegs into uncertain holes using multi-view images and deep network trained on synthetic data, 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp.5792-5799, 2019.
- [4] 吉川恒夫: マニピュレータの力制御, 計測と制御, Vol. 30, No.5, pp.383-388, 1991.
- [5] D. E. Whitney: Quasi-static assembly of compliantly supported rigid parts, Journal of Dynamic Systems, Measurement, and Control, Vol.104, No.1, pp.65-77, 1982.
- [6] T. Inoue, G. D. Magistris, A. Munawar, T. Yokoya, and R. Tachibana: Deep reinforcement learning for high precision assembly tasks, 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp.819-825, 2017.
- [7] S. Scherzinger, A. Roennau, and R. Dillmann: Contact skill imitation learning for robot-independent assembly programming, 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp.4309-4316, 2019.
- [8] S. Gubbi, S. Kolathaya, and B. Amrutur: Imitation Learning for High Precision Peg-in-Hole Tasks, 2020 6th International Conference on Control, Automation and Robotics (ICCAR), pp.368-372, 2020.
- [9] A. Hussein, M. M. Gabar, E. Elyan, and C. Jayne: Imitation Learning: A Survey of Learning Methods, ACM Computing Surveys (CSUR), Vol.50.2, No.21, pp.1-35, 2017.
- [10] S. Levine, C. Finn, T. Darrell, and P. Abbeel: End-to-end training of deep visuomotor policies, The Journal of Machine Learning Research, 17.1, pp.1334-1373, 2016.
- [11] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel: Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation, IEEE International Conference on Robotics and Automation (ICRA), pp.5628-5635, 2018.
- [12] H. Kim, Y. Ohmura, and Y. Kuniyoshi: Using human gaze to improve robustness against irrelevant objects in robot manipulation tasks, IEEE Robotics and Automation Letters, 5.3, pp.4415-4422, 2020.
- [13] H. Kim, Y. Ohmura, Yoshiyuki, and Y. Kuniyosh: Gaze-based dual resolution deep imitation learning for high-precision dexterous robot manipulation, IEEE Robotics and Automation Letters, 6.2, pp.1630-1637, 2021.
- [14] F. Codevilla, M. Muller, A. Lopez, V. Koltun, and A. Dosovitskiy: End-to-end driving via conditional imitation learning, 2018 IEEE International Conference on Robotics and Automation (ICRA), pp.4693-4700, 2018.
- [15] A. Sasagawa, K. Fujimoto, S. Sakaino, and T. Tsuji: Imitation learning based on bilateral control for human-robot cooperation, IEEE Robotics and Automation Letters, 5.4, pp.6169-6176, 2020.
- [16] S. Sakaino: Bilateral Control-Based Imitation Learning for Velocity-Controlled Robot, arXiv:2103.04004, 2021.
- [17] S. Sakaino, K. Fujimoto, Y. Saigusa, and T. Tsuji: Imitation Learning for Variable Speed Object Manipulation, arXiv:2102.10283, 2021.
- [18] Y. Ujitoko and Y. Ban: Survey of Pseudo-haptics: Haptic Feedback Design, and Application Proposals, IEEE Transactions on Haptics, 2021.
- [19] M. Rietzler, F. Geiselhart, J. Gugenheimer, and E. Rukzio: Breaking the tracking: Enabling weight perception using perceivable tracking offsets, Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, pp.1-12, 2018.
- [20] K. L. Palmerius, D. Johansson, G. Höst, and K. Schönborn, An Analysis of the Influence of a Pseudo-haptic Cue on the Haptic Perception of Weight, International Conference on Human Haptic Sensing and Touch Enabled Computer Applications, Springer, Berlin, Heidelberg, pp.117-125, 2014.
- [21] 佐藤理晴, 永田陽子: 疑似触力覚の研究動向と展望, 駒澤大学心理学論集, KARP, 20, pp.29-32, 2018.
- [22] W. Chen and S. Wu: CNN²: Viewpoint Generalization via a Binocular Vision, NeurIPS 2019, ページ, 2019.
- [23] S. James, M. Freese, and A. J. Davison: Pyrep: Bringing v-rep to deep robot learning, arXiv:1906.11176, 2019.
- [24] E. Rohmer, S. P. N. Singh, and M. Freese: Coppeliassim (formerly v-rep): a versatile and scalable robot simulation framework, International Conference on Intelligent Robots and Systems (IROS), 2013.

丹羽靖治: 岐阜大学大学院自然科学技術研究科在学中. 深層模倣学習を用いてロボットアームの自動化を行う研究に従事.

相澤宏旭: 2018 年岐阜大学大学院工学研究科博士前期課程修了. 同年, 同院博士後期課程入学. 現在, 広島大学先進理工系科学研究科助教. 深層学習を用いた画像認識と画像生成の研究に従事. 博士 (工学).

加藤邦人: 1996 年中京大学大学院情報科学研究科修士課程修了. 同年, 同院博士課程入学. 現在, 岐阜大学工学部教授. 2011 年メリーランド大学 Faculty Staff. 画像処理, コンピュータビジョン, 特に深層学習とその応用の研究に従事. 電子情報通信学会, 電気学会, 精密工学会会員. 博士 (情報認知科学)