26th SICE Symposium on Computational Intelligence

July 10, 2025

第 26 回コンピューテーショナル・インテリジェンス研究会

講演論文集

期　日：2025 年 7 月 10 日(木)

会　場：電気通信大学 UEC アライアンスセンター 100 周年記念ホール

主　催：計測自動制御学会　システム・情報部門

企　画：コンピューテーショナル・インテリジェンス部会，知能工学部会

協　賛：システム制御情報学会, 日本知能情報ファジィ学会, 進化計算学会,電気学会, 情報処理学会，日本神経回路学会，日本機械学会，人工知能学会，ヒューマンインタフェース学会，電子情報通信学会， IEEE Computational Intelligence Society Japan Chapter (CISJ), IEEE Systems, Man, and Cybernetics Society Japan Chapter

カタログ番号 25PG0003

公益社団法人計測自動制御学会(SICE)
〒101-0052 東京都千代田区神田小川町 1-11-9 金子ビル 4 階

カタログ番号　２５　ＰＧ　０００３

発行日：2025 年 7 月 10 日
発行者：公益社団法人計測自動制御学会　システム・情報部門
コンピューテーショナル・インテリジェンス部会，知能工学部会

第 26 回コンピューテーショナル・インテリジェンス研究会プログラム

期日：2025 年 7 月 10 日（木）11:00～18:00

会場：電気通信大学 UEC アライアンスセンター 100 周年記念ホール

7 月 10 日(木)

10:30～　　受付開始

11:00　　開会の挨拶

11:15～12:00　招待講演 1　　座長　信川創（千葉工業大学）

言語埋め込み表現による認知構造の計算論的解析と精神医学的応用

小島 大樹（国立精神・神経医療研究センター）

12:00-13:00　運営委員会

13:00-14:00　招待講演 2　　座長　信川創（千葉工業大学）

共同研究が開始される！しまったとならないためにちょっと立ち止まってみませんか？

樋口 人志（あらき知財綜合事務所(HIPP 知財事務所)弁理士

/福井大学客員教授/福井県立大学特命教授）

14:30-15:00　特別講演 1

社会に届く最適解：進化計算と産学連携が切り拓く可能性

佐藤 寛之（電気通信大学）

15:00-15:15　ディスカッション

15:15-15:45　特別講演 2

ニューロモルフィックコンピューティングとロボット

三木 大輔（千葉工業大学）

16:00-18:00　一般セッション　　座長　佐藤 寛之（電気通信大学）

1. スパイキングニューラルネットワークを用いた 深層強化学習における burn-in の適用
○ 岩田尭大，吉岡シャーン圭允，瀧ヶ崎広登，三木大輔（千葉工業大学）

2. オオミズナギドリの生態調査のための動画に基づく群れのサイズと飛行速度の推定
○ 矢頭安樹，井手上哲士，田中彰一郎，畠中利治（福知山公立大学），
本藤聡仁（京都府立西舞鶴高校）

3. 四元数ニューラルネットワークによるロボットアームの逆運動学学習
○ 橋本尚典，礒川悌次郎，上浦尚武(兵庫県立大学)

4. 多目的最適化におけるクラスタリングによるマルチモダルパレートセット推定
○ 鈴村 祐貴(電気通信大学)，太田 恵大(三菱電機株式会社)，佐藤 寛之(電気通信大学)

18:00　　閉会の挨拶

18:30-20:30　懇親会

スパイキングニューラルネットワークを用いた深層強化学習における burn-in の適用

○岩田尭大　吉岡シャーン圭允　瀧ヶ崎広登　三木大輔（千葉工業大学）

Burn-In Strategy for Deep Reinforcement Learning Using Spiking Neural Networks

∗T. Iwata, S. Yoshioka, H. Takigasaki and D. Miki (Chiba Institute of Technology)

Abstract– Deep reinforcement learning offers high performance but with high energy costs. Spiking neural networks (SNNs) improve energy efficiency but suffer instability in DRL. We propose embedding SNNs in TD3 actors with a burn-in strategy inspired by Recurrent Experience Replay in Distributed Reinforcement Learning, achieving improved training stability and higher rewards in OpenAI Gym tasks.

Key Words: Deep Reinforcement Learning, Spiking Neural Networks, TD3, Burn-in Strategy

1　はじめに

近年，深層強化学習（Deep Reinforcement Learning, DRL）アルゴリズムの発展により，Atari ゲームにおいて人間を凌駕する性能が示されている [1]．また，囲碁分野においても同様の成果が報告されている [2]．さらに，ゲーム領域を超えて，DRL は自律移動などのロボット制御タスクにおいて顕著な可能性を示し，多様な環境下で頑健な行動を学習可能である [3]．特に，Twin Delayed Deep Deterministic Policy Gradient（TD3）のような連続値制御アルゴリズムは，観測次元が多く，行動が連続値で定義される環境において，安定かつ効率的なポリシーの学習を可能にしている [4]．しかしながら，これらの手法は大規模な人工ニューラルネットワーク（Artificial Neural Network，ANN）を持ち，学習および推論時の電力消費が大きいことが知られている [5]．このため，バッテリ駆動の自律ロボットや組み込みデバイスへの適用には課題が残されている．そこで，人間の脳に着想を得たスパイキングニューラルネットワーク（Spiking Neural Network，SNN）が有望な代替手段として着目されている．SNN は専用計算素子上で高い電力効率を実現可能であり [6]，Deep Reinforcement Learning with Population-Coded Spiking Neural Network for Continuous Control (PopSAN) においては，アクター・クリティック型 DRL アルゴリズムのアクターネットワークを SNN に置き換えることで，MuJoCo 連続値制御ベンチマークタスク（Ant-v4, Walker2d-v4, Hopper-v4, HalfCheetah-v4）における歩行動作の学習に成功している [7]．また，六足ロボットの歩行制御タスクでは，エネルギー消費をペナルティ化することで，SNN ベースの DRL アルゴリズムはエネルギー効率の高い制御ポリシーを獲得できることが示されている [8].

一方，既存の SNN ベースの DRL アルゴリズムにおいて，学習の不安定さが課題となっている．本研究では，Recurrent Experience Replay in Distributed Reinforcement Learning（R2D2）[9] に着想を得た Burn-in を組み込むことで TD ターゲットの推定精度を向上させ，学習を安定化する手法を提案する．リプレイバッファに蓄積された膜電位に対して，現在のスパイキングアクターネットワークのパラメータに基づき膜電位を再計算し，より正確な TD ターゲットを算出する．本研究の有効性は，MuJoCo 連続値制御ベンチマークタスク（Ant-v4, Walker2d-v4, Hopper-v4, HalfCheetah-v4）において評価し，既存の SNN ベースの DRL 手法と比較して，平均最大報酬および学習の安定性の両面において優れた改善が確認された．

2　手法

本節は，(1) スパイキングアクターネットワーク，(2) SNN の内部状態情報を格納する拡張リプレイバッファ，(3) リプレイバッファから取り出した膜電位の陳腐化問題に対処する Burn-in，および（4）膜電位の探索範囲の縮小による学習効率の向上を目的とした膜電位の量子化処理の四つの要素で構成される．ここで，本研究におけるネットワークの構成を Fig.1 に示す.

2.1　スパイキングアクターネットワーク

スパイキングアクターネットワークは，連続的な環境観測を入力としてスパイク列を生成・処理し，最終的に連続値の行動を出力する．まず，エンコーダモジュールにおいて観測ベクトル s に対して重み付き線形変換と非線形活性化関数を適用する．得られた信号と符号を反転した信号を連結し，複数チャネルでポピュレーションコーディングを行う．この際，一様分布 $U(0,1)$ に従うノイズを付加することで発火を促し，出力されるスパイクの多様性を高める．続いて，スパイキングアクターネットワークにおける SNN モジュールは，三層の全結合 SNN で構成される．各層ではまず，入力スパイク列 $X^l[t]$ を重み行列 W^l とバイアス b^l による線形変換で特徴空間に射影し，以下のように入力電流として表現する

$$I^l[t] = W^l\, X^l[t] + b^l. \tag{1}$$

続いて，Leaky Integrate-and-Fire（LIF）モデルに基づき，時刻 t における膜電位を以下のように定義する

$$U^l[t] = \beta\, U^l[t-1] + I^l[t] - U_{\mathrm{thr}}\, S^l[t-1]. \tag{2}$$

ここで，$U^l[t]$ は時刻 t の膜電位，β は膜電位の減衰率，U_{thr} は発火閾値，$S^l[t-1]$ は前時刻の発火スパイ

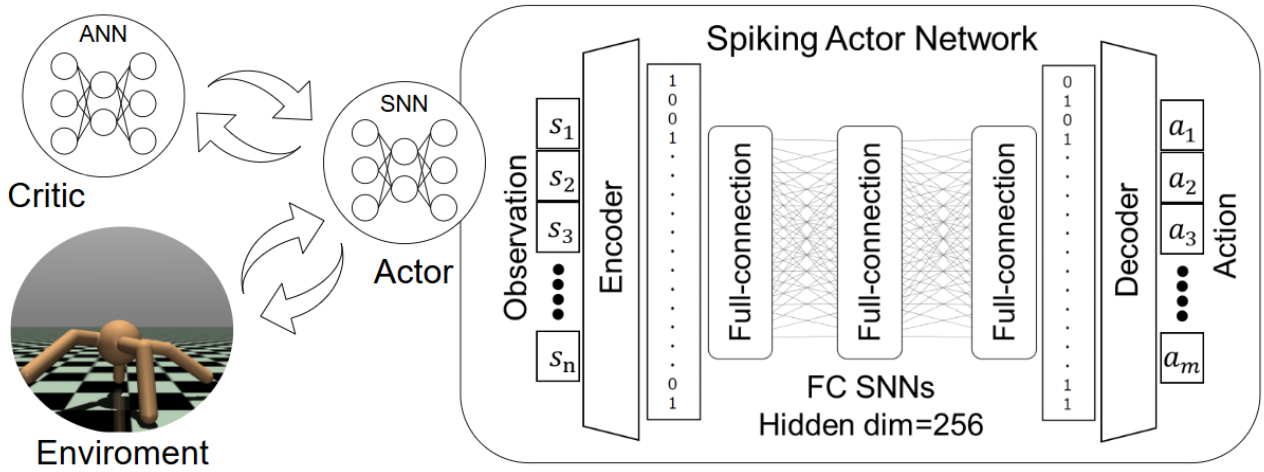


Fig. 1: 提案手法におけるスパイキングアクターネットワークの全体構成図

クである．膜電位が閾値を超えた場合，スパイクを発火し，膜電位をリセットする．この一連の処理を各層で繰り返すことで，SNN モジュールは動的な内部状態を保持しながら時系列に応じたスパイク列を生成する．また，各ステップにおける全層の膜電位は内部状態としてリプレイバッファに保持される．これにより、タイムステップが 1 であるネットワーク構造を持ちながらも，時間的に連続した観測に対する応答の履歴が記録され，時系列情報を活かした行動決定が可能となる．デコーダモジュールでは SNN モジュールから得られるスパイク列を発火率に基づいて集約し，得られた信号に重み付き線形変換と非線形活性化を適用し，連続値の制御信号を得る．

2.2 優先度つき経験再生を用いたリプレイバッファ

従来の DRL 手法では，オフポリシー学習を可能にするために，遷移を (s, a, r, s', d) というタプルとしてリプレイバッファに格納する．本研究では，これを拡張し，SNN の内部状態である膜電位 m を，行動生成前後の両方で保存するために，(s, a, r, m, s', m', d) という形のタプルを格納する．この拡張により，SNN が保持する時間的情報を活用し，より正確な TD ターゲットの算出を可能にする．

さらにリプレイバッファの有効性を高めるため，本手法では Loss Adjusted Prioritized (LAP) [10] を導入する．LAP では，各遷移 i の優先度を絶対 TD 誤差 $\delta(i)$ に基づいて以下のように計算する

$$p(i) = \max\big(|\delta(i)|^{\alpha},\, 1\big). \tag{3}$$

ここで，$\alpha \in (0, 1]$ は優先度が学習に与える影響を制御するハイパーパラメータであり，今回の実験では 0.4 とした．また，リプレイバッファから遷移 i をサンプリングする確率は，次の式で与えられる

$$P(i) = \frac{p(i)}{\sum_{j \in B} p(j)}. \tag{4}$$

ただし，B はリプレイバッファ内のすべての遷移の集合を表す．この優先度付けにより，より大きな TD 誤差を持つ学習に効果的な遷移がより頻繁にサンプリングされるようになる．

2.3 Burn-in の適用

SNN を用いた DRL では，リプレイバッファに蓄積された膜電位がネットワークパラメータの更新に伴って陳腐化するという問題が生じる．この問題に対処するため，本研究では，R2D2 [9] に着想を得た Burn-in を提案する．R2D2 では，Distributed Prioritized Experience Replay [11] に Long Short-Term Memory (LSTM) [12] を統合し，リプレイバッファから取り出した連続する経験の冒頭数ステップを慣らし運転として LSTM に入力することで，その保存された内部状態を現在のネットワークパラメータに適合させた．

本手法では，リプレイバッファから取り出した膜電位を再度スパイキングアクターネットワークに入力し，SNN の内部状態（膜電位）を現在のネットワークパラメータに合わせて更新する．ここで，Burn-in の有無による TD ターゲット計算の流れを Fig.2 に示す．なお，本図は概略であり，ターゲットネットワークの数など一部実装とは異なる点がある．再計算された膜電位 m'_{t+1} は，次式のように得られる

$$m'_{t+1} \leftarrow \pi(s_t, m_t \mid \theta_{\text{now}}). \tag{5}$$

ここで，現在のスパイキングアクターネットワークのパラメータを θ_{now} とする．この Burn-in により，常に最新のスパイキングアクターネットワークによって得られる内部状態に近い情報が学習に利用されるため，TD ターゲットの精度向上と学習の安定性改善が期待できる．

2.4 膜電位の量子化

先行研究に対して，最適なポリシーを得るまでの学習時間が長くなる原因の一つとして，膜電位の取りうる値が連続かつ広範であるために探索範囲が大きくなるという点が挙げられる．そこで本研究では，探索範囲を適度に縮小しつつ性能を維持するために，スパイキングアクターネットワークの順伝搬・逆伝搬中ではなく，リプレイバッファへ保存する前の膜電位にのみ量子化処理を適用する．行動の演算後に得られる膜電位 m をリプレイバッファへ保存する直前に，以下の式を用いて量子化を行う

$$\hat{m}_i = \text{round}\big(m_{\text{clip},i}/\Delta_i\big)\,\Delta_i. \tag{6}$$

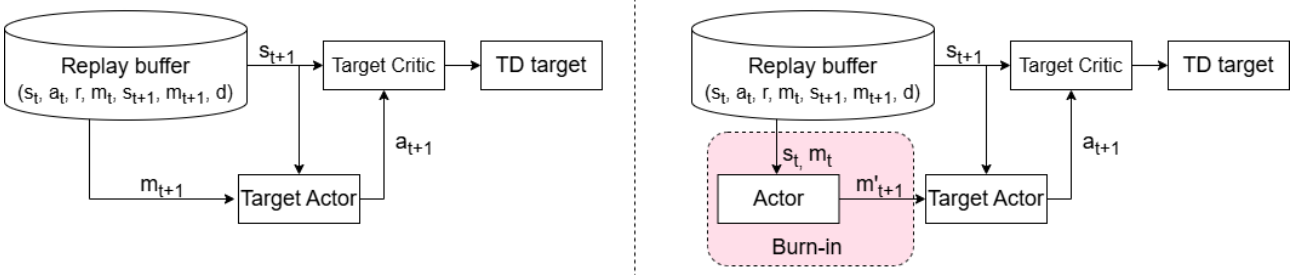


Fig. 2: 一般的な TD ターゲット計算 (左) と Burn-in を導入した場合 (右) の概略図.

ここで，$m_{\mathrm{clip},i}$ は

$$m_{\mathrm{clip},i} = \begin{cases} -2\sigma, & m_i < -2\sigma, \\ m_i, & -2\sigma \le m_i \le 2\sigma, , \\ 2\sigma, & m_i > 2\sigma. \end{cases} \tag{7}$$

であり，Δ_i は，

$$\Delta_i = \begin{cases} 1.0, & |m_{\mathrm{clip},i}| \le \sigma, \\ 2.0, & |m_{\mathrm{clip},i}| > \sigma, \end{cases}, \tag{8}$$

である．また，今回の実験では，$\sigma = 4.0$ とした．こうして得られた量子化膜電位 $\hat{m}$ をリプレイバッファに保存し，学習時には $\hat{m}$ を用いることで，探索範囲を縮小しつつ性能劣化を抑制することを目指す．

3 結果および考察

3.1 実験内容

実験評価には，MuJoCo 連続値制御ベンチマークタスク（Ant-v4, Walker2d-v4, Hopper-v4, HalfCheetah-v4）を使用する．実験は，各手法につき独立した 10 試行を実施する．各試行は 100 エポックからなり，1 エポックは 1 万ステップの学習と 10 回のテストから構成され，これらのテストにおける平均報酬を記録する．Burn-in は学習時のみ適用し，テスト時には使用しない．

本研究では二つの実験を通じて，Burn-in および膜電位の量子化の効果を評価する．実験 1 では，先行研究手法（PopSAN (TD3)）と，Burn-in のみを適用した提案手法を比較した．実験 2 では，Burn-in を適用した提案手法において，膜電位の量子化の有無による性能を比較した．

実装には PyTorch [13] および snnTorch [14] を用いた．スパイキングアクターネットワークおよびクリティックネットワークは，ともに隠れ層が 256 ノードで構成され，学習には Adam [15] を用いた．学習率はそれぞれ 2×10^{-5}，5×10^{-4} とした．スパイキングアクターネットワーク内の SNN では LIF ニューロンを用い，膜電位の減衰率を一様分布 $U(0.1, 0.3)$ に基づいて設定した．また，リプレイバッファサイズは 10^5 である．

3.2 実験結果

Table 1 に，各手法における平均最大報酬を示す．また，平均報酬の推移を Fig.3 および Fig.4 に示す．Fig.3 から，学習後期において，提案手法が先行研究を上回る結果を得られたことが分かる．また，Fig.4 から，Burn-in に加えて膜電位の量子化を併用すると，特に Hopper-v4 において，報酬の収束までに要するステップ数が短縮される結果を得られたことが分かる．

3.3 考察

本研究の結果から，保存された膜電位を現在のスパイキングアクターネットワークのパラメータで再計算することで，リプレイバッファ利用時に生じやすい膜電位の陳腐化が緩和され，行動価値推定誤差が減少し，学習後期においても比較的安定した収束挙動が得られたと考えられる．

膜電位の量子化を Burn-in と併用することで，Hopper-v4 においては，報酬の収束までに要するステップ数が改善された．量子化によりリプレイバッファに格納される膜電位の離散化が進み，探索範囲が適度に縮小された結果，リプレイバッファから取り出した膜電位を利用する際の計算コストを抑制しつつ，最終的な学習性能への影響を最小限に保てたと考えられる．

一方で，Ant-v4，Walker2d-v4，HalfCheetah-v4 では，膜電位の量子化による報酬の収束までに要するステップ数に目立った改善は認められなかった．観測次元数の多いタスクでは，膜電位の量子化による探索範囲の縮小が状態表現全体にほとんど影響を与えず，学習初期の探索効率改善が十分に確認できなかったと考えられる．

4 結論

本研究では，TD3 アルゴリズムにおけるアクターネットワークを SNN で実装し，リプレイバッファを拡張して膜電位を格納・再利用できる仕組みを提案した．さらに，R2D2 に着想を得た Burn-in を導入し，保存された膜電位の陳腐化を抑制するとともに，リプレイバッファ保存前の膜電位に量子化を適用することで，探索範囲を適度に制限し，学習効率の向上を目指した．

MuJoCo 連続値制御ベンチマークタスク（Ant-v4, Walker2d-v4, Hopper-v4, HalfCheetah-v4）を対象とした比較実験の結果，提案手法は既存手法（PopSAN (TD3)）を上回る性能と学習の安定性を示し，平均最大報酬および学習の安定性の両面で一定の改善が得られた．

一方，Burn-in および量子化の適用には追加の計算コストが伴うため，学習全体のステップ数や実行時間が増加するという課題が残されている．今後は Burn-in の再計算ステップ数の最適化や量子化粒度の動的制御を検討し，計算コストと学習性能のさらなる両立を目指していく．また，実機のロボットを用いた評価も実施し，消費電力などの実環境下での制御性能を検証していく予定である．

参考文献

1) Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wier-

Table 1: 各手法ごとの平均最大報酬

Task	PopSAN (TD3)	Ours (Burn-in)	Ours (Burn-in + Quantization)
Ant-v4	5234±865	5997±107	6103±87
Walker2d-v4	4456±895	5187±258	5188±492
Hopper-v4	3439±105	3509±180	3697±97
HalfCheetah-v4	10208±1359	10257±338	10650±249

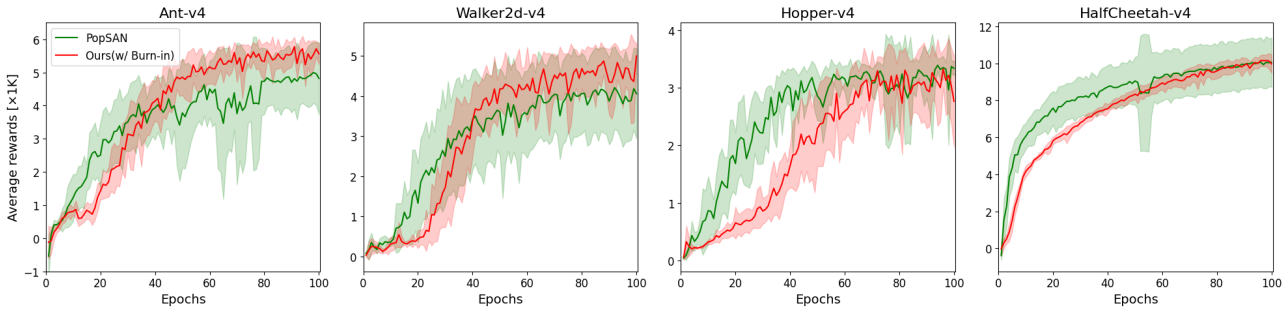


Fig. 3: PopSAN (TD3) と Burn-in を適用した提案手法の平均報酬の推移

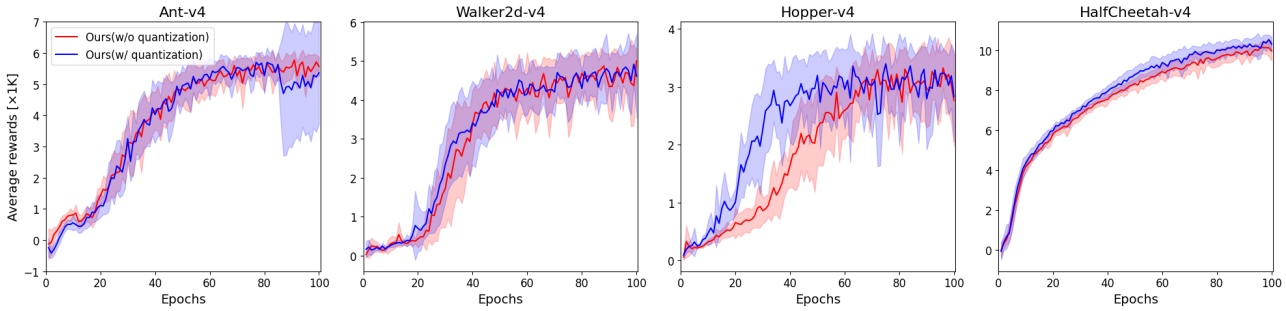


Fig. 4: 提案手法における膜電位の量子化の有無による平均報酬の推移

stra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013. URL: `http://arxiv.org/abs/1312.5602`, https://arxiv.org/abs/1312.5602 `arXiv:1312.5602`.

2) David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

3) Tuomas Haarnoja, Sehoon Ha, Aurick Zhou, Jie Tan, George Tucker, and Sergey Levine. Learning to walk via deep reinforcement learning. In Antonio Bicchi, Hadas Kress-Gazit, and Seth Hutchinson, editors, *Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019*, 2019.

4) Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

5) Mark Horowitz. 1.1 computing ' s energy problem (and what we can do about it). In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pages 10–14. IEEE, 2014.

6) Intel Corporation. Neuromorphic computing - next generation of ai, 2022. URL: `https://www.intel.com/content/www/us/en/research/neuromorphic-computing.html`.

7) Guangzhi Tang, Neelesh Kumar, Raymond Yoo, and Konstantinos Michmizos. Deep reinforcement learning with population-coded spiking neural network for continuous control. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 2016–2029. PMLR, 2021. 16–18 Nov 2021.

8) Katsumi Naya, Kyo Kutsuzawa, Dai Owaki, and Mitsuhiro Hayashibe. Spiking neural network discovers energy-efficient hexapod motion in deep reinforcement learning. *IEEE Access*, 9:150345–150354, 2021.

9) Steven Kapturowski, Georg Ostrovski, Will Dabney, John Quan, and Remi Munos. Recurrent experience replay in distributed reinforcement learning. In *International Conference on Learning Representations*, 2019.

10) Scott Fujimoto, David Meger, and Doina Precup. An equivalence between loss functions and non-uniform sampling in experience replay. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc.

11) Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver. Distributed prioritized experience replay. In *International Conference on Learning Representations*, 2018.

12) Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

13) PyTorch Foundation. Pytorch, 2022. URL: `https://pytorch.org/`.

14) Jason K Eshraghian. snntorch, 2021. URL: `https://snntorch.readthedocs.io/en/latest/snntorch.html`.

15) PyTorch Foundation. Adam, 2022. URL: `https://pytorch.org/docs/stable/generated/torch.optim.Adam.html`.

オオミズナギドリの生態調査のための動画に基づく群れのサイズと飛行速度の推定

○矢頭安樹（福知山公立大学大学院）　井手上哲士　田中彰一郎　畠中利治（福知山公立大学）
本藤聡仁（京都府立西舞鶴高等学校）

Estimating Flock Size and Flight Speed based on Ecological Research Video of Streaked Shearwaters

∗A. Yazu, S. Ideue, S. Tanaka and T. Hatanaka (The University of Fukuchiyama),
A. Hondo (Kyoto Prefectural Nishimaizuru Senior High School)

Abstract− In this study, we propose a detection system for monitoring flocks of Streaked Shearwaters around Kanmuri Island in Maizuru City. We examine two different approaches for analyzing video data collected by the Kanmuri Island Ecological Survey Group. The first approach utilizes a deep learning-based method, employing the well-known YOLO framework for target object detection and tracking. The second approach is based on background subtraction between consecutive frames to identify flying objects in the scene. Additionally, we estimate the flying speed of the detected objects to infer the size and movement characteristics of the flocks. The estimated speeds are consistent with the known flight speed of Streaked Shearwaters, supporting the validity of the proposed method.

Key Words: Streaked Shearwater, object detection, target tracking, flock size estimation, flight speed estimation.

1　はじめに

京都府舞鶴市の若狭湾内に位置する冠島は，西日本最大のオオミズナギドリ繁殖地として知られ，国の天然記念物に指定されている．オオミズナギドリは全長約 50cm ほどの海鳥であり，1965 年には京都府の鳥に指定されている．太平洋やインド洋を生活の場としており，日本には繁殖のために飛来する．主に離島に営巣し島の周辺の広い範囲の海域で餌を採取することから，海洋生態系における高次捕食者に位置付けられている．このため，海洋生態系の健全性を評価する指標としても注目されている．このような背景からオオミズナギドリの生態に関する研究は国際的にも進められており，日本でもいくつかの飛来地における生態調査が実施されている．また，オオミズナギドリは繁殖開始年齢が遅く，1 年に 1 つの卵しか産まないという特徴を持つ．このため，生息数が減少するとその回復が困難である．全国的に準絶滅危惧種に指定されており，本稿で対象にしている冠島が所在する京都府では、府の改訂版レッドリスト 2021 に掲載され要注目種に指定されている [1]．今後も適切な繁殖環境を維持していくために，生態学的な基礎情報の収集が不可欠とされている．さらに，冠島のオオミズナギドリは，沖合で餌を探し繁殖地に戻るときに島の周りを旋回する「鳥まわり」と呼ばれる行動をする．この行動の解析も個体の移動パターンなど生態研究において重要な意味を持つ．

我々は，このような調査の一環として，京都府立西舞鶴高等学校理数探究科が実施している「オオミズナギドリ調査を通した地域協働型探究学習と普及啓発活動 [2]」との協働を進め，調査隊が撮影した「鳥まわり」の映像を用いて，YOLO を活用したオオミズナギドリの検出と追跡，速度推定を行った．また，いわゆる深層学習を用いる計算負荷が高い手法ではなくエッジで撮影した動画が処理できることを目標に，負荷が小さい処理として背景差分法に基づくオオミズナギドリの検出と追跡，速度推定に取り組んでいる．本稿では，これらの 2 つの手法による結果を紹介する．2 章では YOLOv8 を用いた検出と追跡について紹介し，追跡結果からの移動速度の推定結果を示す．また，3 章では背景差分法に基づく検出を行い，画像上での移動速度から実空間における移動速度の推定を行った結果を紹介し，4 章でまとめと今後の展望を述べる．

2　YOLO を用いた物体検出・追跡

2.1　YOLO の概要

YOLO(You Only Look Once) [3] は画像に映っている物体を検出し，認識するための深層学習に基づくアルゴリズムである．このような画像から対象の物体を検出し認識する手法は古くから研究されている．基本的な方法論は，検出対象の物体のテンプレートを用意し，対象とする画像の局所領域とテンプレートのパターンマッチングにより適合度を求め，対象の存在を把握するというものである．したがって，従来の検出法は対象の画像の調べたい領域のサイズに応じた検出窓をスライドさせる仕組みを必要としていた．これに対して，YOLO は，検出窓をスライドさせる仕組みを用いることなく画像を 1 回，畳み込みニューラルネットワーク（CNN）へ通すことでオブジェクトの検出と認識が可能である．多くの層をもつニューラルネットワークに，検出窓の大きさを変更しつつ画像上をスライドさせる処理とパターンマッチングのための畳み込み計算が埋め込まれており，ネットワークを構成する素子と結合重みに多数のテンプレートに相当する情報が埋め込まれていることになる．学習済のネットワークには多くの検出対象の情報が埋め込まれているが，新しい対象物体に対しては，追加学習を行うことで対応ができる．ひとたび学習したネットワークは，対象の物体の検出のための処理が比較的高速で，入力動画像から

一度で複数の物体の検出が可能である．また，検出した物体についてバウンディングボックス，クラス名，信頼度を出力する機能を有する．鳥類への適用についての先行研究 [4, 5] では，特定の種（カワウ）に対する検出や，ビオトープへ飛来する鳥類の検出と飛行軌跡の認識に利用されている．本研究では，冠島のオオミズナギドリに固有の「鳥まわり」の様子を捉えた動画に適用し，個体の検出と追跡を行う．撮影するカメラは，陸地（島の内部）に設置され，遠方を飛行する群れの様子を撮影している．視野内に対象の個体が存在する時間が比較的短く，一定の方向への群れ行動である．

YOLO による物体検出の仕組みのイメージを Fig. 1 に示す [?]．

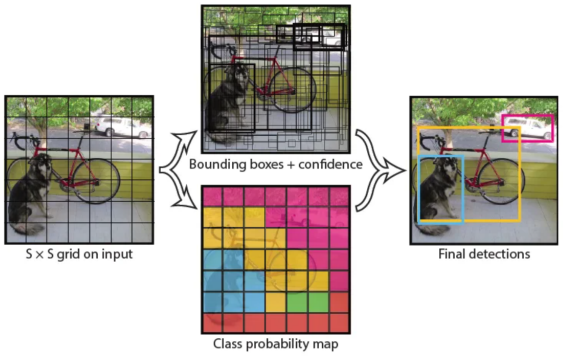


Fig. 1: YOLO の仕組み．技術紹介サイト [6] より引用

まず，Fig.1 の左側に示されている処理では，物体を検出したい画像を正方形にリサイズした後，細かい正方形グリッドに分ける．次に，中央に示す処理では，各グリッドセルにおいて検出された物体の中心位置と高さ，幅を推定してバウンディングボックスで囲む．また，線の太さは信頼度を表し，太ければ太いほど高い信頼度を持つことを表している．中央下の処理では，各グリッドセルで検出された物体のクラスごとに色分けして分類している．これらの「物体の位置検出」と「物体のクラス分類」の処理を組み合わせることで，最終的な検出結果が出力されている．

本研究では，物体検出において YOLOv8 を使用した．YOLOv8 は，2023 年に Ultralytics 社が公開した YOLO のバージョンである．YOLOv8 は，最先端のバックボーンとネックアーキテクチャを採用し，特徴抽出と物体検出の性能を向上させている．また，精度と推論速度のバランスが最適化されているとされ，リアルタイム物体検出を必要とする多様な応用分野で高い有用性をもつとされている．さらに，YOLOv8 は複数の事前学習済みモデルを提供し，用途や性能要件に応じた柔軟なモデル選択が可能である．なお，本研究で用いた YOLOv8 は 2023 年に発表されているが，2025 年 6 月現在はバージョン 12 が発表されている．

YOLOv8 では，検出されるクラス名に開発者が収集した事前学習済みのデータセット (COCO データセット) を利用することができる．COCO(Common Objects in COntext) データセットは，物体認識やセグメンテーション，キャプション生成などのタスクに広く用いられる大規模な画像データセットである．COCO データセットには 33 万枚の画像が収録されており，20 万枚の画像には物体検出，セグメンテーション，キャプションのタスクのためのアノテーションが付けられている．COCO

Table 1: COCO データセットの一部

ID	ラベル
0	人 (person)
1	自転車 (bicycle)
2	自動車 (car)
3	オートバイ (motorcycle)
4	飛行機 (airplane)
5	バス (bus)
6	列車 (train)
7	トラック (track)
8	ボート (boat)
9	信号機 (traffic light)
10	消火栓 (fire hydrant)
11	停止標識 (stop sign)
12	駐車メーター (parking meter)
13	ベンチ (bench)
14	鳥 (bird)
15	猫 (cat)
16	犬 (dog)

データセットの一部を Table 1 に示す．人や自転車，車などの 80 種類のデータセットが学習されている．

ここでは，PyTorch を用いてモデルを実装し，学習回数 100 回で学習を行った．評価指標として平均適合率（mAP）を用い，検出性能を評価した．この YOLOv8 を Anaconda Prompt の仮想環境上で pip を使用しインストールした後に実装し，物体検出を実施した．出力された結果を Fig. 2 に示す．Fig. 2 を見ると，奥の岩は boat(ボート)，鳥は people(人) と認識されている．また，検出されていない鳥が多い．既存のデータセットと学習モデルでは物体が認識されないため，独自の学習データセットを作成した．

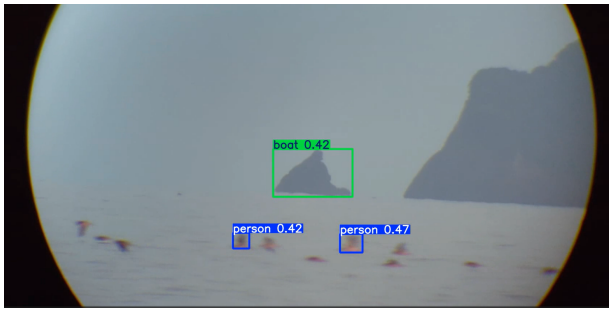


Fig. 2: 出力結果

2.2 独自の学習データ

YOLOv8 には，演算負荷と精度の異なる 5 つの学習済み重みモデルがあり，「n，s，m，l，x」の 5 つが存在する．しかし，YOLOv8 に Git で用意されているこれらの学習モデルでは，動画内の物体が綺麗に検出されなかったため，LabelImg を利用して独自にラベルと学習データを作成した．

学習用に用意した画像 500 枚に対してアノテーションを行い，bird（鳥）と rock（画像上真ん中の岩）の 2 つのラベルを作成した．ここでは，ラベリングされたバウンディングボックスの座標がテキストファイルで保存されている．学習には，YOLOv8 のモデルの中で精度と速度のバランスが取れた YOLOv8m を使用した．また，学習回数は 300 回と設定した．学習されて

Fig. 3: 画像の出力結果

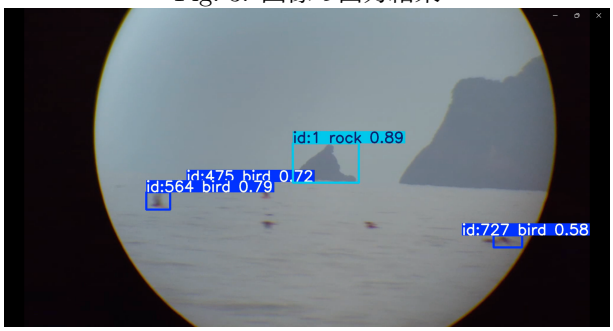


Fig. 4: 動画の出力結果の一部

できた独自のデータセットを使用して最初と同じ動画と画像の物体検出を行った．

このとき得られた物体検出結果の一例を Fig. 3, 4 に示す．Fig. 3 を見ると，学習データとして設定したラベル通りにクラス分けが行われ，正確に検出されているのがわかる．しかし，Fig. 4 の結果では，飛行している鳥の検出結果にばらつきがみられる．この理由として，信頼度と閾値が適切ではないということが挙げられる．そこで，信頼度と閾値を変更し，検出結果にばらつきが生じなくなるように試行錯誤して調整した．

信頼度とは，[0, 1] の範囲の実数で表され，画像上のバウンディングボックスで囲まれた部分が「背景」か「物体」かを与える数値のことである．信頼度が 0 に近いほど，バウンディングボックスの中身は「背景」である可能性が高く，信頼度が 1 に近いほど，バウンディングボックスの中身は検出対象の「物体」を表しているということになる．なお，YOLOv8 が提供しているデフォルト値は 0.5 である．検出結果の精度を上げるために，0.6，0,65，0.7 などと変更することによって，この値以上の値の信頼度を持ったバウンディングボックスだけを出力するようにする必要がある．

また，NMS (Non-Maximum suppression) 閾値を利用し，各クラスごとに，一番信頼度の高い枠を選び，その枠と他の枠の重なり具合 (IoU) を調べて一定以上の割合で重なっている枠を削除していく．IoU 閾値とは，基準の枠とその他の枠の重なり具合（割合）を表す数字のことで，信頼度と同様に [0, 1] の範囲の実数で示される．IoU 閾値が 0 に近いほど，2 つの枠は「重なっていない」を意味し， IoU 閾値が 1 に近いほど，バウンディングボックスの中身は「重なっている」を意味する．IoU のデフォルト値は 0.45 で，重複を減らすにはこの値を上げる必要がある．信頼度と同じように，IoU も 0.5，0.6，0.65 と値を変更して出力させた．

信頼度と閾値を変更した出力結果の一例を Fig.5 に

Fig. 5: 信頼度:0.7, 閾値:0.65

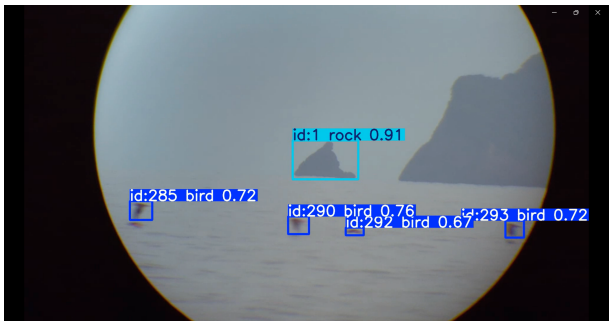


Fig. 6: 信頼度:0.6, 閾値:0.5

示す．Fig. 5 は，信頼度 0.7，閾値 0.65 とどちらも高い値を適用して出力させた結果の画像である．どちらの数値も高くしすぎるとバウンディングボックスが検出されなくなった．このように，独自の学習データを用いて画像を出力する際，信頼度や閾値を任意の値に設定し，10 回の検証を行った．この結果，Fig. 6 に示す信頼度 0.6，閾値 0.5 の設定のときに満足できる正確さをもつバウンディングボックスが出力できることが確認された。

追加した対象物体を表す bird も rock もバウンディングボックスが重複することなく，1 つのバウンディングボックスとして適切に検出できた．しかし，左側の鳥のうち一羽が認識されていない．これは，信頼度が低いために背景として認識された可能性や，閾値の影響で両端の物体のバウンディングボックスと重ならず検出されなかった可能性が考えられる．

今後の課題として，信頼度と閾値を調整し，未検出の鳥がどの条件で認識されるかを確認する必要がある．また，未検出の原因が物体の特徴や環境要因（背景とのコントラスト，波の反射などによる光の影響など）によるものかを分析し，適切なパラメータ設定やデータの追加が有効かを検討する必要がある．

2.3 追跡

本研究では，YOLOv8 にトラッキング機能を統合することで，単なるフレームごとの物体検出にとどまらず，時間的連続性を考慮した物体追跡を実現するリアルタイム検出追跡システムを構築した．本システムでは，YOLOv8 によって各フレームで検出された物体に対し，SORT（Simple Online and Realtime Tracking）や Deep SORT（Deep Simple Online and Realtime Tracking），あるいは ByteTrack などのオンラインマルチオブジェクトトラッキングアルゴリズムを組み合わせることで，各物体に一意の ID を付与し，連続したフレーム間で同一物体を追跡する．

2.4 MOT(Multiple Object Tracking) フォーマット

MOT フォーマットは，物体追跡タスクにおけるデータの標準的な形式で，このフォーマットは，物体が動画や画像のシーケンス内でどのように動いているか，追跡される過程を表現する．MOT フォーマットは，各フレームごとに物体の位置や識別情報を記録する形式で，通常，以下のようなカンマ区切りのテキストファイル形式で保存される．今回は，フレーム番号，オブジェクト ID，バウンディング左上の x 座標，バウンディングボックス左上 y 座標，バウンディングボッスの幅，バウンディングボックスの高さ，信頼度がカンマ区切りで出力されている．後半の 2 つは，フォーマット上で無視する値である．MOT フォーマットを利用して出力されたテキストファイルの一部を Table 2 に示す．

Table 2: オブジェクトの移動速度データ

F	ID	X	Y	W	H	Label	Score
1	1	912.86	454.41	223.26	133.68	rock	0.90
1	2	1325.18	717.48	64.51	45.10	bird	0.73
1	3	1439.16	793.20	58.34	23.22	bird	0.68
1	4	1166.85	713.80	54.63	26.70	bird	0.68
1	5	852.07	745.11	95.28	32.05	bird	0.59
1	6	1676.83	733.55	53.34	52.20	bird	0.44
2	1	913.52	454.59	223.87	133.43	rock	0.90
2	2	1305.05	716.21	60.40	39.72	bird	0.66
2	4	1147.78	710.87	50.89	27.29	bird	0.69
2	5	857.40	746.25	67.26	27.82	bird	0.48
2	6	1655.16	734.11	53.31	55.80	bird	0.69
3	1	914.27	455.49	223.79	132.84	rock	0.90
3	2	1281.81	713.65	59.66	35.09	bird	0.69
3	4	1126.87	708.79	49.66	27.60	bird	0.68
3	6	1634.57	732.44	57.77	59.01	bird	0.77
4	1	914.36	456.46	223.04	132.69	rock	0.90
4	2	1256.75	710.51	54.86	31.29	bird	0.72
4	4	1102.45	706.44	51.05	29.78	bird	0.71
4	6	1610.08	731.47	63.33	58.19	bird	0.75
5	1	914.84	456.51	222.66	132.53	rock	0.89
5	2	1227.56	707.01	55.88	32.10	bird	0.74
5	4	1076.38	703.07	55.72	34.66	bird	0.75
5	6	1588.30	731.29	62.69	52.54	bird	0.72
6	1	914.67	456.52	222.51	132.44	rock	0.89
6	2	1200.29	702.50	55.91	34.79	bird	0.74
6	4	1052.35	699.51	56.70	40.24	bird	0.74
6	6	1560.92	730.31	64.64	41.37	bird	0.71
6	22	737.76	743.04	86.09	26.27	bird	0.53
7	1	913.48	456.13	223.77	132.41	rock	0.90
7	2	1172.61	700.32	57.00	35.64	bird	0.72

これによって出力された物体のデータを基に各フレーム間での物体の移動速度を計算した．まず，各フレーム間のバウンディングボックスの中心座標の変化量を求め，その変化量と各フレーム間の時間差を使用して移動距離と移動速度を計算している．単位は，pixel/s，m/s，km/h の 3 パターン出力されるようになっている．このテキストファイルでは，オブジェクト ID，信頼度，物体が検出されたフレーム ID，画面上での移動速度 (pixels/s)，推定速度 (m/s,km/h) が出力される．オオミズナギドリはおおよそ時速 30km で飛行するとされている．この追跡法で出力されたテキストファイルの結果からは，時速 40〜50km となっており，差はあるが，速度推定の結果は安定した値を得ていることがわかる．速度推定を行い，テキストファイルにまとめたものの一部を Table 3 に示す．

最後に，MOT フォーマットで出力されたテキストファイルからバウンディングボックスの中心座標を計算し，各オブジェクト ID の位置変化をグラフで可視化した．x 軸は，”Frame Number” でフレーム番号を設定し時間の経過を示している．y 軸は”Position (Pixels)” で中心座標をピクセル単位 (物体の位置座標) で表示している．位置変化のグラフを Fig. 7 に示す．

Fig. 7 の軌跡の傾きがオオミズナギドリの飛行速度

Table 3: オブジェクトの推定移動速度

Obj ID	Class	Frame	Center (X, Y)	px/s	km/h
1	rock	2	(1025.45, 521.30)	57.48	1.94
2	bird	2	(1335.25, 736.07)	1340.20	45.23
4	bird	2	(1173.22, 724.51)	1255.12	42.36
5	bird	2	(891.03, 760.16)	519.45	17.53
6	bird	2	(1681.82, 762.01)	1297.22	43.78
1	rock	3	(1026.16, 521.91)	55.47	1.87
2	bird	3	(1311.64, 731.19)	1433.71	48.39
4	bird	3	(1151.70, 722.59)	1285.20	43.38
6	bird	3	(1663.45, 761.95)	1091.88	36.85
1	rock	4	(1025.88, 522.80)	55.86	1.89
2	bird	4	(1284.18, 726.15)	1660.32	56.04
4	bird	4	(1127.98, 721.33)	1412.91	47.69
6	bird	4	(1641.74, 760.57)	1293.70	43.66
1	rock	5	(1026.17, 522.77)	17.34	0.59
2	bird	5	(1255.50, 723.06)	1715.50	57.90
4	bird	5	(1104.24, 720.40)	1412.60	47.68
6	bird	5	(1619.64, 757.56)	1326.38	44.77
1	rock	6	(1025.92, 522.74)	14.72	0.50
2	bird	6	(1228.24, 719.89)	1631.75	55.07

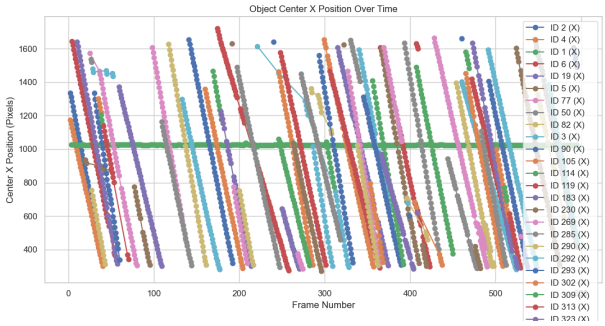


Fig. 7: 各オブジェクトの位置関係

に対応している．y 軸の値がほぼ一定の緑で示した軌跡は画像上で動いていないものに対応し，背景に移る岩である．Fig.7 を見ると，ある区間では鳥の位置が連続して検出され，直線的な動きが確認できる．一方で，点がまばらになっている部分もあり，これは検出や追跡が一時的に途切れてしまったためだと考えられる．より安定した検出を行うには，検出精度の向上に加えて，追跡アルゴリズムの強化も重要である．特に，ID が正しく継続して付与されるような工夫が必要であると考える．

3 背景差分による追跡

2 章では，深層学習に基づく YOLO v8 を用いた検出について述べた．この方法は，撮影条件に応じたセッティングも必要になるため，計算資源が期待できない冠島[1]における現地調査の際に活用することは期待できない．そこで，固定カメラによる撮影であることおよび冠島を周回する「鳥まわり」を撮影していることを前提に，その動画に写っている移動物体を検出することに特化した背景差分法を用いたオオミズナギドリの検出，追跡および飛行速度の推定について検証を行った．

3.1 差分画像

差分画像とは，動画における近接する画像間の差分をとることで得られる．同一の背景をもつ 2 枚の画像の差分を求めると，同じピクセル座標における画素の色や明るさの差があった場合にその部分だけが残り，画像上に表示される．動画におけるフレームごとの差を求めることによって，画角上で移動した物体を抽出することが可能になる．ここでは，動画中の動体（鳥）の出現位置を可視化するために，OpenCV の背景差分ア

[1]冠島は無人島でありワークステーション級の計算機を使うことができない環境である．

ルゴリズムである MOG2（Mixture of Gaussians）を用いた前景抽出処理を実施した．オオミズナギドリが「鳥まわり」を行っている動画から一定間隔（任意のフレーム数ごと）でフレームを抽出し，それぞれのフレームに対して背景差分を適用する．差分画像を 2 値化し，動きのある部分だけを白く表示した画像を保存した．処理結果の一例として，3 フレーム，10 フレームごとの背景差分処理結果を Fig.8，Fig.9 に示す．検出されたオオミズナギドリの飛行軌跡が白の領域で示されている．

Fig. 8: 3 フレームごと

Fig. 9: 10 フレームごと

このように，鳥が飛行しているときの軌道や，翼の動きがわかりやすく表示されているのがわかる．しかし，どのようなスピードで動いているかなど鳥の速度や動画内に写る鳥の数などの情報は，このままでは分からない．

3.2　差分画像による鳥の追跡

差分画像を用いて，動画内における鳥が 1 フレームに動くピクセル距離や 1 秒間に動く距離を計算し出力することに取り組んだ．鳥の動画内における速度を求めるため，背景差分で出力されたオオミズナギドリの領域を楕円で近似し，その中心点を求めて，楕円の中心点間の距離（ピクセル距離）を測ることとした．これは，はばたきの影響で検出されるオオミズナギドリの領域のサイズが変化するため，その領域を近似する楕円の中心を採用することで，ロバストな移動量の計算ができることを期待したものである．楕円の中心点が 1 フレームごとに移動するピクセル数を計算し，動画のフレームレートに基づいてオオミズナギドリの動画内における移動速度（pixel/s）を計算した．楕円でオオミズナギドリが検出された領域を近似した結果を Fig. 10 に示す．

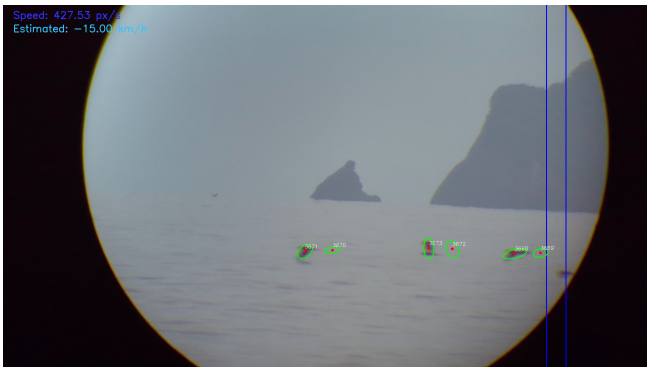


Fig. 10: 処理結果

Fig. 10 では，元フレームとその 3 フレーム後の差分から得たオオミズナギドリの存在領域の楕円を，元の画像に重ねた画像である．実際の処理は，2 値化した画像データのもとで処理されていることに注意されたい．なお，Fig. 10 においてオオミズナギドリがいない部分にも楕円が存在するのは，元フレームから 3 フレーム後のフレームとの差分を表示しているためである．

次に，差分対象の楕円による近似とフレーム内における鳥の速度を利用して，動画内に写る鳥の数と鳥が実際に移動しているおおよその速度を計算して出力した．

次に，動画全体で検出されたオオミズナギドリの個体数をカウントする方法を示す．撮影対象は，群れをなして「鳥まわり」と呼ばれる島を周回する行動であるため，動画では画面の右から左へほぼ一定の速度で飛行していることを前提とできる．そこで，最初のフレームに写る鳥の数をまずカウントし，2 フレーム以降は，画面の右からある範囲のみの検出区間を設定し，そこに現れたオオミズナギドリのみをカウントすればよい．検出区間をフレームあたりの移動量をもとに決定することができるため，仮に撮影条件が異なっても，当初の複数のフレームから速度を求め，検出範囲を決定することができる．すなわち，フレーム間差分と画像の右側の狭い領域における対象の検出のみで，動画全体に移る個体数の推定が可能である．Fig. 10 に示した動画の場合は，2 フレーム以降は横方向のピクセル数で，1620 から 1678 の間の 59 ピクセル間の領域に注目し，この範囲の楕円の中心の個数を調べることで，飛行するオオミズナギドリの個体数がカウントできる．

また，オオミズナギドリの飛行速度は約 30km/h であるということから，この動画上の平均的な移動速度を 30km/h と仮定して，フレーム差分から求めた画像上の移動速度 (pixel/s) を換算し飛行速度とした[2]．得られたフレーム間の速度推定と，動画内の鳥の数をカウントした結果を Table 4 に示す．

Table 4 は検出した個体の速度情報の一例であるが，すべての検出されたオオミズナギドリの速度情報から，2 フレーム以降にカウントすべき領域を横方向で 1620 ピクセルから 1678 ピクセルの矩形領域に制限し，個体数をカウントする実験を行ったところ，最初のフレームに 7 羽，第 2 フレーム以降で 75 羽の計 82 羽であっ

[2]撮影場所から群れが飛行している領域までの距離情報が得られていなかったため，pixel/s から実空間における速度への幾何学的な換算は行っていない．飛行領域の奥行きもあるため，すべての個体がほぼ一定とみなせる速度で移動していたとしても，画像上の速度はある範囲に分布する．こういった点も踏まえると画像からの速度の推定には 2 章の結果も含めて，さらなる検討が必要である

Table 4: オブジェクト間の速度情報

From ID	To ID	Speed (px/s)	Speed (km/h)
1	3	2045.26	29.87
1	9	1984.81	29.27
4	5	2046.51	29.89
5	8	2176.06	31.18
6	7	2064.10	30.06
6	10	2290.00	32.32
8	10	2107.25	30.49
9	11	1749.96	26.92

た．実際に目視でコマ送りしながらカウントした結果は 80 羽であったので，この方法で実用上問題ない精度で個体数を調べることができると考えられる．

4　まとめ

YOLOv8 を用いた物体検出では，信頼度と閾値を変更することで比較的正確に 1 つのバウンディングボックスで検出することができた．また，物体追跡のために YOLOv8 に tracking 機能を追加し，オオミズナギドリの ID 検出とフレーム間の物体を識別し，物体の速度の推定を行った．さらに，MOT フォーマットを出力することで，フレーム ID やクラス名，バウンディングボックスの情報を基に，対象のオオミズナギドリが飛んでいる動画の各フレーム間の対象の移動距離と，撮影場所や島と対象物の距離間などから，誤差はあるが鳥の速度を測定することができた．最後に各オブジェクトの位置関係を可視化することで，フレームが進むごとにオブジェクト ID がほぼ一定速度で同じ方向への移動をしていることが容易に確認できる出力を得た．

差分画像に基づいてオオミズナギドリのフレーム間の移動距離を算出し，画像上における鳥の速度に基づいて，画像内で個体数をカウントする領域を制限できることを示した．背景差分による個体の検出と，初期データに基づいて個体数をカウントする領域を制限する手法は，小型の省電力型の計算機での実用に適している．冠島は無人島であるが，携帯電話回線は利用できる．したがって，ソーラー発電とバッテリーで安定稼働できる程度の小型の計算機を用いた，動画の記録から個体数のカウントまで行える計測システムが稼働すると，計測結果をほぼリアルタイムで確認できるようなシステム化が期待できる．今後，検出精度の改善とともに動画の取得から処理結果の出力までの処理手順の計算負荷の削減などを行い，現地で稼働するシステムを試作することを今後の課題としたい．

謝辞

本プロジェクトで対象をした動画は，2024 年 8 月に冠島調査研究会とともに島を訪れて，鳥まわり行動の動画を撮影した西舞鶴高等学校理数探究科の冠島調査グループから提供されたものです．無人島である冠島での昼夜逆転の調査活動のたまものであり，関係者の皆さまをはじめ多くの方の協力があって実現しました．その多大な努力に感謝します．また，本研究の一部は西舞鶴高等学校の令和 7 年度高等学校 DX 加速化推進事業の経費から補助を受けたものです．ここに謝意を表します．

参考文献

1) 京都府改訂版レッドリスト 2021（哺乳類・鳥類）, https://www.pref.kyoto.jp/kankyo_red/news/, (2021), 2025 年 6 月 30 日閲覧

2) 京都府立西舞鶴高等学校, 日本鳥類保護連盟 Web サイト, https://www.jspb.org/, 2025 年 6 月 30 日閲覧

3) Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi : You only look once: Unified, real-time object detection, Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788(2016)

4) 北風裕教, 吉原蓮人, 岡部蒼太, 松村遼 : オブジェクト検出 YOLO を用いた害鳥認識システムの開発, 産業応用工学会論文誌, Vol. 8, No. 1, pp. 10–16 (2020)

5) 中山紘喜, 安枝裕司, 久冨学, 野上敦嗣, 松本亨 : 深層学習法を用いた鳥類の動画自動検出追跡システムの開発と響灘ビオトープでの運用, 環境共生, Vol. 39, No.1, pp. 45–54(2023)

6) kindamu24005,【YOLO】の仕組みを簡単にまとめてみた【物体検出アルゴリズム】, Qiita,2023 年 9 月 24 日, https://x.gd/v6gD8, 2025 年 6 月 30 日閲覧

四元数ニューラルネットワークによるロボットアームの逆運動学学習

○橋本尚典　礒川悌次郎　上浦尚武（兵庫県立大学）

On Learning Inverse Kinematics for Robotic Arms with Quaternionic Neural Networks

∗T. Hashimoto, T. Isokawa and K. Naotake (University of Hyogo)

Abstract− This study examines whether Quaternionic Neural Networks (QNNs) are suitable for learning inverse kinematics mappings from joint rotation inputs. Although QNNs are theoretically well-suited for representing 3D rotations, their practical effectiveness in this context remains unclear. To investigate this, we use a 6-DoF UR5e robotic arm and Jacobian data collected in MuJoCo to compare QNNs with standard real-valued neural networks in approximating the Jacobian matrix. Experimental results did not show significant performance improvements with QNNs. Nonetheless, this study provides empirical insights into the limitations and challenges of using quaternion-based representations in rotation-related tasks in robotics and contributes to a better understanding of model design for periodic input data.

Key Words: Neural Network, Inverse kinematics, Quaternion

1　はじめに

近年の AI モデルの成功は, 対象タスクに応じた構造をモデル内部に適切に組み込むことに大きく依存している. たとえば, 畳み込みニューラルネットワークは画像認識に適した局所構造を持ち, Transformer は長い時系列や自然言語処理に適した自己注意機構を備えることで, 高い性能と効率的な学習を実現している. 本研究が着目するのは,「ロボットの関節角データを入力とするタスクにおいて, どのようなモデル構造が最も適しているか」という問いである. これは依然として明確な解が得られていない課題である.

この問題設定の具体例として, 本研究では逆運動学に着目する. 逆運動学とは, ロボットの手先を所望の位置・姿勢に到達させるために必要な関節角度を求める問題であり, ロボットアームの運動制御における基本かつ重要な課題である. ロボット構造が複雑になるにつれて逆運動学の解析的解法は困難となり, 一般には数値的な反復手法に依存することが多い. こうした数値解法では, ロボットの運動学モデルから導出されるヤコビ行列を用いた最適化アルゴリズムが一般的に利用される. ヤコビ行列を用いる手法は高い精度を実現する一方, その導出や実装は煩雑であり, MuJoCo [7] や Pinocchio [1] などのソフトウェアにより計算支援が行われている. しかし, これらのソフトウェアは実行環境に制約があり, とくにエッジデバイスのような軽量な計算環境には適さないという課題がある.

そこで本研究では, ヤコビ行列の計算をニューラルネットワーク (Neural Network; NN) により近似し, より軽量かつ効率的に逆運動学を解く手法を検討する. また, この枠組みの中で, 角度情報を入力とするタスクにおいて, より適したネットワーク構造を探索することを目的とする. とくに, 回転のような周期的構造をもつデータに対しては, その幾何学的性質を反映した適切な表現が求められる. 従来の NN では, 回転を単なる実数ベクトルとして扱うことが多く, その本質的な構造が考慮されていない場合がある. こうした課題に対処するために, 本研究では四元数ニューラルネットワーク (Quaternionic Neural Network; QNN) [5, 3] に着目する. QNN は, 四元数演算を NN の計算に組み込んだモデルであり, ロボット分野への応用例も報告されている [6, 2]. 四元数は三次元空間の回転を表現する数学的枠組みであり, ジンバルロックの回避や滑らかな補間が可能であることから, ロボティクスにおいて広く利用されている. 本研究では, 各関節の回転を四元数で表現し, それを入力とする QNN を構築することで, 回転の構造をより自然に捉え, ヤコビ行列の近似精度と汎化性能の向上を図る. 数値実験では, 6 自由度のロボットアームである UR5e を対象とし, MuJoCo 上で取得したヤコビ行列データをもとに, 各関節の回転を四元数で表現した入力と, 対応するヤコビ行列を出力とする QNN を構築する. さらに, 同条件下で通常の実数値 NN と比較し, 学習誤差の収束性および近似精度の観点からその有効性を検証する.

2　逆運動学の数値解法

まず, ロボットアームにおける逆運動学の数値的解法として, ヤコビ行列を用いた微分逆運動学に基づくアプローチについて説明する. 勾配降下法は, 目的関数の勾配に沿って変数を更新し, 関数の極小値を探索する最適化手法である. この手法を逆運動学に適用することで, 目標とする手先の位置および姿勢に対応する関節角を, 逐次的に数値的に求めることが可能となる.

ここで, 手先の目標状態を表すベクトルを $\boldsymbol{s}_{\text{goal}} \in \mathbb{R}^6$ とする. $\boldsymbol{s}_{\text{goal}}$ は, 手先の 3 次元位置および 3 次元姿勢を表すベクトルである. 現在の関節角を $\boldsymbol{q}_k \in \mathbb{R}^n$ とし, 順運動学関数を $\boldsymbol{\phi} : \mathbb{R}^n \to \mathbb{R}^6$ と定義する. この関数 $\boldsymbol{\phi}(\boldsymbol{q}_k)$ は, 関節角 $\boldsymbol{q}_k$ に対してロボットの手先の位置と姿勢を出力する. このとき, 手先の目標状態に関する残差は次のように定義される：

$$\boldsymbol{e}(\boldsymbol{q}_k) = \boldsymbol{s}_{\text{goal}} - \boldsymbol{\phi}(\boldsymbol{q}_k) \tag{1}$$

ヤコビ行列 $\boldsymbol{J}(\boldsymbol{q}_k) \in \mathbb{R}^{6\times n}$ は次式で定義される：

$$\boldsymbol{J}(\boldsymbol{q}_k) = \frac{\partial \boldsymbol{\phi}(\boldsymbol{q})}{\partial \boldsymbol{q}}\Big|_{\boldsymbol{q}=\boldsymbol{q}_k} \tag{2}$$

このとき, 関節角の更新則は次式で与えられる：

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k - \alpha \boldsymbol{J}(\boldsymbol{q}_k)^{\mathrm{T}} \boldsymbol{e}(\boldsymbol{q}_k) \tag{3}$$

ここで $\alpha > 0$ は学習率である．この更新則により, 手先の目標位置・姿勢と現在の位置・姿勢の差を最小化する方向に関節角を更新することで, 所望の手先状態を実現する関節角を逐次的に求めることができる．

3　四元数ニューラルネットワーク

四元数ニューラルネットワーク (Quaternionic Neural Network; QNN) [5, 3] は, 入力・出力の各成分を四元数として表現し, それに対する線形変換をハミルトン積に基づいて定義するものである．まず, 四元数について説明し, その後 QNN を構成する要素について説明する．

3.1　四元数

四元数は, 複素数を拡張した超複素数の一種であり, 1 つの実数成分と 3 つの虚数成分からなる 4 次元の数体系である．四元数 $x \in \mathbb{H}$ は以下のように定義される：

$$x = x^{(r)} + i\, x^{(i)} + j\, x^{(j)} + k\, x^{(k)}, \tag{4}$$

ここで $x^{(r)}, x^{(i)}, x^{(j)}, x^{(k)} \in \mathbb{R}$ は各成分を表し, i, j, k は以下の関係を満たす虚数単位である：

$$i^2 = j^2 = k^2 = ijk = -1. \tag{5}$$

また, i, j, k の間には次のような関係が成立する：

$$\begin{aligned} ij &= k, & jk &= i, & ki &= j, \\ ji &= -k, & kj &= -i, & ik &= -j. \end{aligned}$$

四元数の積 xy は以下のように計算される：

$$\begin{aligned} xy = {} & (x^{(r)}y^{(r)} - x^{(i)}y^{(i)} - x^{(j)}y^{(j)} - x^{(k)}y^{(k)}) \\ & + i\,(x^{(r)}y^{(i)} + x^{(i)}y^{(r)} + x^{(j)}y^{(k)} - x^{(k)}y^{(j)}) \\ & + j\,(x^{(r)}y^{(j)} - x^{(i)}y^{(k)} + x^{(j)}y^{(r)} + x^{(k)}y^{(i)}) \\ & + k\,(x^{(r)}y^{(k)} + x^{(i)}y^{(j)} - x^{(j)}y^{(i)} + x^{(k)}y^{(r)}) \end{aligned}$$

なお, 四元数の積は交換則を満たさない $xy \neq yx$.

このような構造は, 3 次元空間における回転の表現に特に有用であり, ロボット工学, コンピュータグラフィックス, ゲーム開発など, 幅広く活用されている．

3.2　線形層

N 次元の四元数ベクトル $\boldsymbol{x}$ を入力とし, M 次元の四元数ベクトル $\boldsymbol{y}$ を出力とする, 四元数 NN の線形層について説明する．

入出力の四元数ベクトル $\boldsymbol{x} \in \mathbb{H}^N, \boldsymbol{y} \in \mathbb{H}^M$ は, 4 つの実数ベクトル $\boldsymbol{x}^{(\cdot)} \in \mathbb{R}^N$, $\boldsymbol{y}^{(\cdot)} \in \mathbb{R}^M$ を要素にもつ：

$$\begin{aligned} \boldsymbol{x} &= \begin{bmatrix} \boldsymbol{x}^{(r)} & \boldsymbol{x}^{(i)} & \boldsymbol{x}^{(j)} & \boldsymbol{x}^{(k)} \end{bmatrix}^{\top}, \\ \boldsymbol{y} &= \begin{bmatrix} \boldsymbol{y}^{(r)} & \boldsymbol{y}^{(i)} & \boldsymbol{y}^{(j)} & \boldsymbol{y}^{(k)} \end{bmatrix}^{\top}. \end{aligned} \tag{6}$$

次に学習で調整する対象となる, 重み行列 $\boldsymbol{W}$ およびバイアスベクトル $\boldsymbol{b}$ も四元数で構成される．

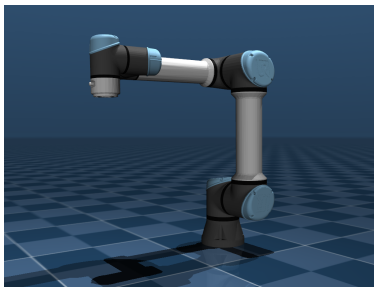


Fig. 1: UR5e robot arm used in this study.

は, 各成分に対応する 4 つの実数行列から構成され, 次のように表される：

$$\boldsymbol{W} = \boldsymbol{W}^{(r)} + i\,\boldsymbol{W}^{(i)} + j\,\boldsymbol{W}^{(j)} + k\,\boldsymbol{W}^{(k)}, \tag{7}$$

$$\boldsymbol{W}^{(r)}, \boldsymbol{W}^{(i)}, \boldsymbol{W}^{(j)}, \boldsymbol{W}^{(k)} \in \mathbb{R}^{n_{\mathrm{out}} \times n_{\mathrm{in}}}. \tag{8}$$

バイアスベクトル $\boldsymbol{b} \in \mathbb{H}^{n_{\mathrm{out}}}$ も同様に, 次のように定義される：

$$\boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}^{(r)} & \boldsymbol{b}^{(i)} & \boldsymbol{b}^{(j)} & \boldsymbol{b}^{(k)} \end{bmatrix}^{\top}. \tag{9}$$

このとき, 出力は以下のようにハミルトン積 $\otimes$ に基づいて定義される線形変換として与えられる：

$$\boldsymbol{y} = \boldsymbol{W} \otimes \boldsymbol{x} + \boldsymbol{b}. \tag{10}$$

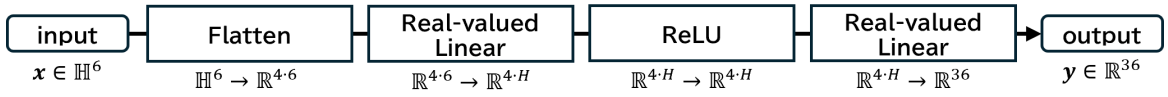
ハミルトン積は四元数の非可換な積に対応するため, これを効率的に実装するためには, 各成分ごとに展開された実数の行列演算形式に変換する必要がある．その具体形は次のように与えられる：

$$\begin{aligned} \boldsymbol{y}^{(r)} &= \boldsymbol{W}^{(r)}\boldsymbol{x}^{(r)} - \boldsymbol{W}^{(i)}\boldsymbol{x}^{(i)} \\ &\quad - \boldsymbol{W}^{(j)}\boldsymbol{x}^{(j)} - \boldsymbol{W}^{(k)}\boldsymbol{x}^{(k)} + \boldsymbol{b}^{(r)} \\ \boldsymbol{y}^{(i)} &= \boldsymbol{W}^{(i)}\boldsymbol{x}^{(r)} + \boldsymbol{W}^{(r)}\boldsymbol{x}^{(i)} \\ &\quad + \boldsymbol{W}^{(k)}\boldsymbol{x}^{(j)} - \boldsymbol{W}^{(j)}\boldsymbol{x}^{(k)} + \boldsymbol{b}^{(i)} \\ \boldsymbol{y}^{(j)} &= \boldsymbol{W}^{(j)}\boldsymbol{x}^{(r)} - \boldsymbol{W}^{(k)}\boldsymbol{x}^{(i)} \\ &\quad + \boldsymbol{W}^{(r)}\boldsymbol{x}^{(j)} + \boldsymbol{W}^{(i)}\boldsymbol{x}^{(k)} + \boldsymbol{b}^{(j)} \\ \boldsymbol{y}^{(k)} &= \boldsymbol{W}^{(k)}\boldsymbol{x}^{(r)} + \boldsymbol{W}^{(j)}\boldsymbol{x}^{(i)} \\ &\quad - \boldsymbol{W}^{(i)}\boldsymbol{x}^{(j)} + \boldsymbol{W}^{(r)}\boldsymbol{x}^{(k)} + \boldsymbol{b}^{(k)}. \end{aligned} \tag{11}$$

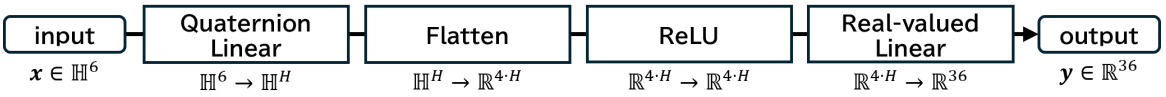
このように, 四元数 NN の線形層は複雑なハミルトン積をあらかじめ実数の演算に展開することで, 通常の深層学習ライブラリ上でレイヤとして定義するだけで, 誤差逆伝播に対応した学習をそのまま適用することができる．

4　数値実験

四元数ニューラルネットワーク（QNN）の有効性を検証するため, 6 自由度のロボットアームを対象に, 関節角度からヤコビ行列を予測する回帰問題を設定し, 数値実験をおこなう．



(a) Neural Network (NN)



(b) Quaternionic Neural Network (QNN)

Fig. 2: Model architectures used in the numerical experiments.

4.1 データ生成とタスク設定

学習および評価データは, 物理シミュレータ MuJoCo 上に構築した UR5e モデル（Fig. 1）を用いて収集した. 具体的には, 関節角度ベクトル $\mathbf{q} \in \mathbb{R}^6$ に対して, MuJoCo の内蔵関数を用いて対応するヤコビ行列 $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{6\times 6}$ を数値的に計算し, その平坦化ベクトル $\mathbf{y} \in \mathbb{R}^{36}$ を教師信号として記録した. このようにして得られた関節角とヤコビ行列の組を用いて, ニューラルネットワークによるヤコビ行列の近似性能を評価した.

4.2 モデル構成

本研究では比較対象として 2 種類のニューラルネットワークアーキテクチャを構築した. Fig. 2 (a) はニューラルネットワーク（NN）, Fig. 2 (b) は四元数ニューラルネットワーク（QNN）の構造を示している. NN モデルでは, 入力として与えられる関節回転データ $x \in \mathbb{H}^6$ をまず実数空間 $\mathbb{R}^{4\times 6}$ に展開し, これを平坦化して $\mathbb{R}^{24}$ のベクトルとする. その後, 実数値の全結合層により中間表現 $\mathbb{R}^{4H}$ を得て, ReLU 活性化関数を適用し, さらに出力層を通じて最終的に $y \in \mathbb{R}^{36}$ を生成する. 一方, QNN モデルでは, 入力 $x \in \mathbb{H}^6$ に対して四元数線形層を適用し, 四元数空間 $\mathbb{H}^H$ に変換する. その後, 得られた出力を実数空間 $\mathbb{R}^{4H}$ に変換・平坦化し, NN と同様に ReLU および実数値の全結合層を経て最終出力を得る. 両モデルとも, 最終的にヤコビ行列を平坦化した 36 次元の実数ベクトルとして出力する構成となっており, 損失関数には出力と教師信号との L2 ノルムを用いて学習を行った.

4.3 入力エンコーディング

ロボットの関節状態は通常, 関節角として表現されるが, 本研究ではこれを四元数に変換してモデルに入力する. 具体的には, 関節角ベクトル $\boldsymbol{q} = [q_1, q_2, \ldots, q_6]^\top \in \mathbb{R}^6$ を, 各関節の回転軸 $\boldsymbol{a}_i \in \mathbb{R}^3$ まわりの回転とみなし, 対応する四元数表現を以下の式により求める：

$$\boldsymbol{x}_i = \begin{bmatrix} \cos(q_i/2) \\ \boldsymbol{a}_i \sin(q_i/2) \end{bmatrix} \in \mathbb{H}. \tag{12}$$

この変換をすべての関節に適用することで, 四元数からなる入力ベクトル $\boldsymbol{x} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_6] \in \mathbb{H}^6$ を得る.

4.4 学習設定

実験では, 学習率 $\eta \in \{10^{-4}, 10^{-5}, 10^{-6}\}$ および隠れ層のユニット数 $H \in \{100, 200, 500\}$ を変化させ, 両モデルの性能を比較した. 最適化手法には Adam [4] を用いた. バッチサイズは 2048, 最大学習ステップ数は 10^6 とし, 各条件について異なる 10 通りの乱数シードを用いて学習を実施した. 学習データは, 各ステップにおいて 2048 個のデータサンプルを逐次生成し, オンライン形式で学習に用いた.

4.5 結果と考察

Fig. 3 に，各ハイパーパラメータ条件での検証損失の推移を示す．行は学習率 $\eta \in \{10^{-4}, 10^{-5}, 10^{-6}\}$，列は隠れ層ユニット数 $H \in \{100, 200, 500\}$ に対応している．赤線は実数値ネットワーク（NN），青線は四元数ネットワーク（QNN）の平均損失を表し，半透明の帯はそれぞれの標準偏差である．学習率 $\eta = 10^{-4}$（上段）では，NN が QNN より速く損失を低下させる傾向が顕著であり，特に $H = 100, 200$ の条件では終盤まで NN が一貫して低い損失を維持した．一方で $H = 500$ では両モデルの曲線がほぼ重なるまで収束し，モデル容量が増えるにつれて差が縮まる様子が確認できる．学習率 $\eta = 10^{-5}$（中段）では，初期収束は依然として NN が優勢だが，学習が進むにつれて両モデルの損失差は縮小し，$H = 500$ では最終損失がほぼ一致した．学習率 $\eta = 10^{-6}$（下段）では，両モデルともに緩やかな収束挙動を示し，全ての H において最終損失はほぼ同一水準に達した．最終ステップにおける平均検証損失を総合的に比較すると，いずれの条件でも一方が他方を明確に上回る傾向は認められなかった．すなわち，ヤコビ行列の回帰という本タスクでは，四元数表現が理論的に回転情報を保持できるにもかかわらず，損失最小化の観点で必ずしも優位性を発揮しないことが示唆される．今後は，より深い QNN アーキテクチャの探索や，四元数演算に適した正則化・初期化手法との併用を検討し，四元数構造の帰納的バイアスを効果的に活かすネットワーク設計指針の確立が求められる．

5 まとめ

本研究では, 関節角からヤコビ行列を予測する回帰問題に対して, 四元数ニューラルネットワーク（QNN）の有効性を検証した. 具体的には, MuJoCo シミュレータ上に構築した 6 自由度ロボットアームモデルを用い, 広範な関節角度サンプルに対応するヤコビ行列データを収集し, QNN と従来の実数値ニューラルネットワーク（NN）との比較を行った.

QNN は回転の構造的特徴を内部に保持したまま学習できる点で理論的に有利とされるが, 数値実験の結果, 学習の収束速度や最終的な予測精度において, NN に対する明確な優位性は確認されなかった. 特に, 学習率が大きい条件下では, QNN の学習がやや不安定になる傾向も見られた. これらの結果は, QNN の表現能力が本タスクにおいて必ずしも直接的な性能向上に結びつくとは限らず, モデル構造とタスク特性との整合性が

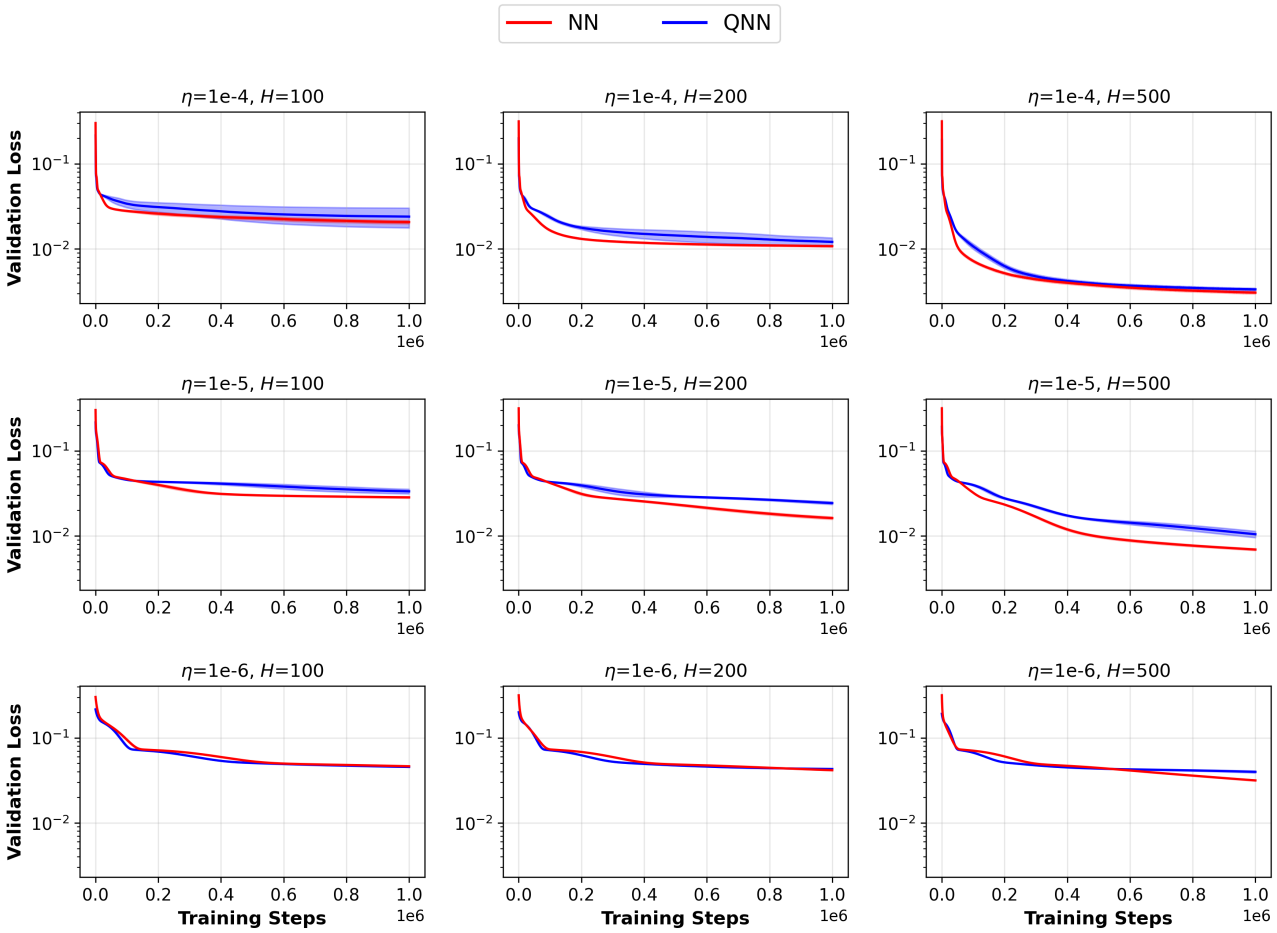


Fig. 3: Validation loss curves of the Neural Network (red) and Quaternion Neural Network (blue) under different learning rates η and hidden layer sizes H. Each plot shows the average over 10 trials with different seeds.

極めて重要であることを示唆している.

今後の課題としては, より深層のネットワーク構造の導入, 四元数演算に適した正則化手法の検討, およびロバスト性や汎化性能を評価するための実ロボット環境での応用実験を通じて, QNN の有効性を引き出すための設計指針の確立を目指す必要がある. あわせて, 周期的・構造的な性質を持つ入力データに対する学習手法として, QNN を理論的に再評価する枠組みの構築も今後の重要な課題である.

参考文献

1) Justin Carpentier, Guilhem Saurel, Andrea Del Prete, Sébastien Tonneau, and Nicolas Mansard. Pinocchio: fast forward and inverse dynamics for poly-articulated systems. In *Proceedings of the IEEE International Symposium on System Integrations (SII)*, pages 439–446. IEEE, 2019.

2) Y. Cui, K. Takahashi, and M. Hashimoto. Design of control systems using quaternion neural network and its application to inverse kinematics of robot manipulator. In *Proceedings of the 2013 IEEE/SICE International Symposium on System Integration (SII)*, pages 527–532, Kobe, Japan, 2013.

3) Teijiro Isokawa, Haruhiko Nishimura, and Nobuyuki Matsui. Quaternionic multilayer perceptron with local analyticity. *Information*, 3(4):756–770, 2012.

4) Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2015. arXiv:1412.6980.

5) N. Matsui, T. Isokawa, H. Kusamichi, F. Peper, and H. Nishimura. Quaternion neural network with geometrical operators. *Journal of Intelligent & Fuzzy Systems*, 15(3-4):149–164, 2004.

6) K. Takahashi. Remarks on control of robot manipulator using quaternion neural network. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 560–565, Honolulu, HI, USA, 2018.

7) Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5026–5033. IEEE, 2012.

多目的最適化におけるクラスタリングによるマルチモダルパレートセット推定

○鈴村 祐貴 (電気通信大学), 太田 恵大 (三菱電機株式会社), 佐藤 寛之 (電気通信大学)

Pareto Set Estimation Using Clustering for Multi-Objective Multimodal Optimization Problems

∗ Yuki Suzumura (The University of Electro-Communications), Yoshihiro Ohta (Mitsubishi Electric Corporation) and Hiroyuki Sato (The University of Electro-Communications)

Abstract– In multi-objective optimization problems, multimodality often arises, where multiple distinct solutions in the decision space correspond to the same point in the objective space, making accurate Pareto set (PS) estimation challenging. This paper proposes a novel PS estimation method that applies clustering to known solutions in the decision space and independently constructs local response surfaces for each cluster. By selectively estimating PS using these cluster-specific models based on direction vectors, the proposed method can capture multiple modes that conventional approaches fail to represent. Experiments using standard multimodal benchmark problems (MMF1–8 and LIRCMOP1–2) demonstrate that the proposed method achieves higher estimation accuracy than conventional methods and effectively suppresses the generation of dominated solutions.

Key Words: Multi-objective optimization, multimodal problem, Pareto set estimation, clustering, response surface modeling

1 はじめに

実世界の最適化問題の多くは，相反する複数の目的関数を含む多目的最適化問題に分類される [1, 2]．一般に，多目的最適化問題には単一の最適解は存在せず，代わりに，目的関数間の最適なトレードオフを表す解集合であるパレートセット（Pareto Set, PS）が存在する．多目的最適化の究極的な目標は，PS 全体を獲得することであるが，現実的には困難である．問題によっては，PS の一部を見出すことすら容易ではなく，PS 全体を網羅するには，膨大な数の解を生成・評価する必要がある．また，実世界の最適化問題においては，1 つの解の目的関数値を算出するために多大な計算コストを要することも少なくない．このような場合には，得られた限られた解から有用な知見を引き出す工夫が求められる．その手段のひとつに，PS 推定（Pareto Set Estimation）[5] がある．

PS 推定は，既知の各解について，目的空間における方向と変数空間における位置関係を，応答曲面によって構築するものである．この方法は，変数空間における PS の構造を推測する有効な手段である．しかし，PS 推定は，目的空間の各方向に対応する変数空間における位置を出力するため，1 つの目的関数値ベクトルが複数の変数値ベクトルに対応するようなマルチモダル問題においては，推定される PS の精度が著しく低下するという課題がある．

本稿では，マルチモダル問題における PS 推定の精度向上を目的として，クラスタリングを活用した新たな PS 推定法を提案する．提案法では，変数空間上の既知の解集合をクラスタリングし，各クラスタごとに独立して PS 推定を行う．本手法の有効性を検証するために，マルチモダルなテスト問題である MMF1–8 [6] および LIRCMOP1–2 [7] を用いて実験する．

2 多目的最適化

変数ベクトル $\boldsymbol{x} = (x_1, x_2, \ldots, x_D) \in \mathbb{R}^D$ に対して M 種類の目的関数 f_i $(i = 1, 2, \ldots, M)$ を内包する多目的最適化問題は，次のように定式化される．

$$\text{Minimize } \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x})) \quad (1)$$

一般的に，各目的関数が相反するため，すべての目的関数値を同時に最小化する解は存在しない．複数の目的間の最適なトレードオフであるパレートフロント (Pareto front, PF) を表すパレート最適解集合 (Pareto set, PS) を獲得することが目標である．

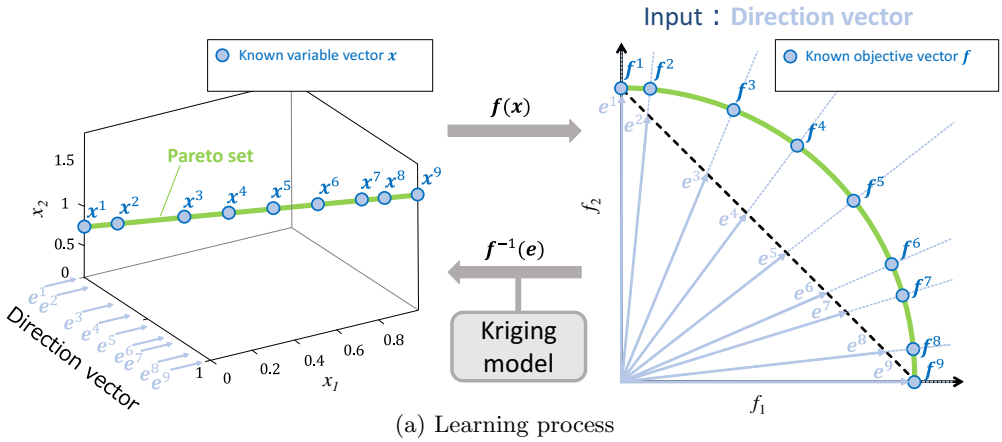
PS のサイズは，問題によるものの無限の可能性がある．進化計算による多目的最適化 [1, 2] では，限られた数の解集合を出力することで PS を表現しようとする．ただ，特に $\boldsymbol{f}(\boldsymbol{x})$ の算出コストが高い場合，多数の解を獲得することに難しさが生じる．その結果，少数の解によって，サイズが無限の可能性がある PS を表現せざるを得なくなる．当然ながら，少数の解では，PS の表現精度は落とさざるを得ず，意思決定者が満足できない結果になり得る．

3 従来法：PS 推定

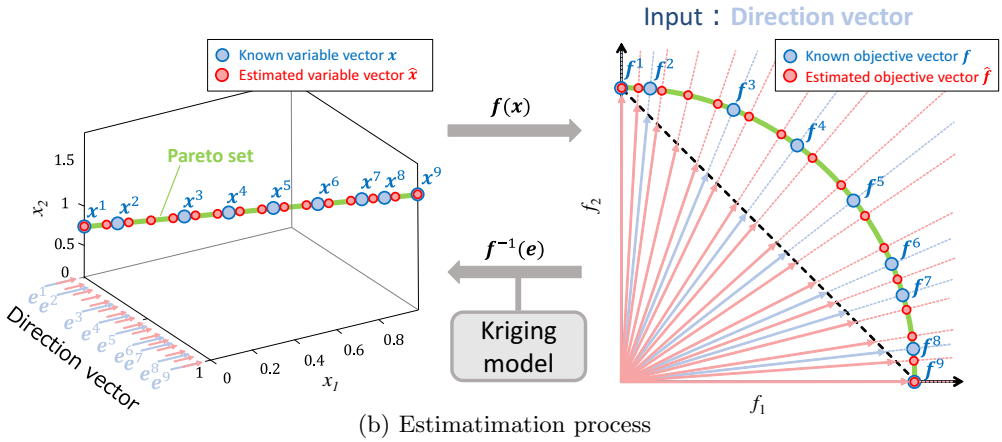
3.1 概要

PS 推定は，少数の既知の解から PS を推定する [5]．PS 推定は，**Fig. 1** に示す 2 つの過程の **(a)** 学習と **(b)** 推定に分かれる．左が変数空間，右が目的空間である．青色の各既知解 $(\boldsymbol{x}^i, \boldsymbol{f}^i)$ $(i \in \{1, 2, \ldots, N\})$ は，変数値ベクトル $\boldsymbol{x}^i$ と目的関数値ベクトル $\boldsymbol{f}^i$ で構成され，変数空間と目的空間の間で一対の関係になる．**Fig. 1** は，$N = 9$ の例である．また，各目的関数値ベクトル $\boldsymbol{f}^i$ を指す方向ベクトル $\boldsymbol{e}^i$ を矢印で示した．

Fig. 1 (a) における学習過程では，既知解集合 $\mathcal{P} = \{(\boldsymbol{x}^1, \boldsymbol{f}^1), (\boldsymbol{x}^2, \boldsymbol{f}^2), \ldots, (\boldsymbol{x}^N, \boldsymbol{f}^N)\}$ について，方向ベクトル群 $\mathcal{E} = \{\boldsymbol{e}^1, \boldsymbol{e}^2, \ldots, \boldsymbol{e}^N\}$ を入力，変数値ベクトル群



(a) Learning process



(b) Estimatimation process

Fig. 1: Conventional Pareto set estimation [5]

$\mathcal{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N\}$ を出力とする応答曲面 $\boldsymbol{x}$-model を Kriging [3, 4] によって求める．

Fig. 1 (b) における推定過程では，赤矢印で示す膨大な方向ベクトル群 $\hat{\mathcal{E}} = \{\hat{\boldsymbol{e}}^1, \hat{\boldsymbol{e}}^2, \ldots\}$ を応答曲面 $\boldsymbol{x}$-model に入力し，推定変数ベクトル群 $\{\hat{\boldsymbol{x}}^1, \hat{\boldsymbol{x}}^2, \ldots\}$ を得ることができる．

変数空間における位置 $\boldsymbol{x}^i$ から目的空間の位置 $\boldsymbol{f}(\boldsymbol{x}^i)$ を求める目的関数 $\boldsymbol{f}$ に対して，PS 推定は，目的空間における方向 $\hat{\boldsymbol{e}}$ から変数空間における位置 $\hat{\boldsymbol{x}}$ を求める逆関数 $\boldsymbol{f}^{-1}$ を構築することになる．

3.2 方法

疑似コードを **Algorithm 1** に示す．入力は，既知の解集合 $\mathcal{P} = \{(\boldsymbol{x}^1, \boldsymbol{f}^1), (\boldsymbol{x}^2, \boldsymbol{f}^2), \ldots, (\boldsymbol{x}^N, \boldsymbol{f}^N)\}$ である．既知の解集合 $\mathcal{P}$ から，1 行目は目的ベクトル群 $\mathcal{F}$，2 行目は変数ベクトル群 $\mathcal{X}$ を取り出す．4–6 行目は，既知の各解の目的ベクトル $\boldsymbol{f}^i$ の方向ベクトル $\boldsymbol{e}^i$ を算出し，既知解集合の方向ベクトル群 $\mathcal{E}$ に加える操作を繰り返す．8 行目は，既知解の方向ベクトル群 $\mathcal{E}$ を入力，変数ベクトル群 $\mathcal{X}$ を出力とする応答曲面の Kriging モデル $\boldsymbol{x}$-model を構築する．これは，**Fig. 1 (a)** において，目的空間における既知解の方向ベクトル (青矢印) に対応する変数空間における位置 $\boldsymbol{x}$(青丸) を学習する処理に該当する．9 行目は，生成する解集合 $\mathcal{Q}$ を初期化している．10–14 行目は，一様分布する膨大な方向ベクトル群 $\hat{\mathcal{E}}$ における各方向 $\hat{\boldsymbol{e}}$ に注目しながら解 $\hat{\boldsymbol{x}}$ を生成する．これは，**Fig. 1 (b)** において，目的空間における任意の方向ベクトル (赤矢印) に対応する変数空間

Algorithm 1 Conventional Pareto Set Estimation [5]

Require: Known solution set $\mathcal{P} = \{(\boldsymbol{x}^1, \boldsymbol{f}^1), (\boldsymbol{x}^2, \boldsymbol{f}^2), \ldots, (\boldsymbol{x}^N, \boldsymbol{f}^N)\}$, a large unit vector set $\hat{\mathcal{E}} = \{\hat{\boldsymbol{e}}^1, \hat{\boldsymbol{e}}^2, \ldots\}$
Ensure: Popoulation $\mathcal{P}'$

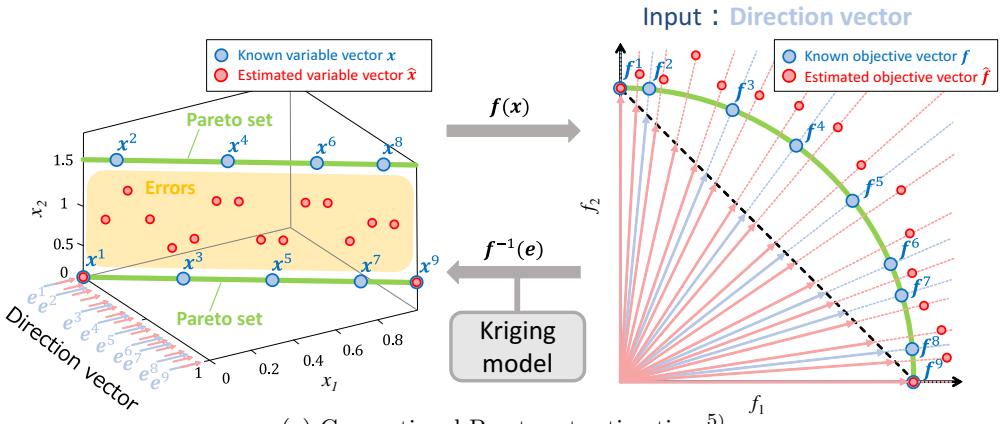
Learning Process

1: $\mathcal{F} = \{\boldsymbol{f}^1, \boldsymbol{f}^2, \ldots, \boldsymbol{f}^N\} \leftarrow$ Extract objective vectors $(\mathcal{P})$
2: $\mathcal{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N\} \leftarrow$ Extract variable vectors $(\mathcal{P})$
3: $\mathcal{E} \leftarrow \emptyset$ ▷ Direction vectors of known solutions
4: **for** $i \leftarrow 1, 2, \ldots, N$ **do**
5: $\quad \boldsymbol{e}^i \leftarrow \boldsymbol{f}^i / \|\boldsymbol{f}^i\|_1$ ▷ Direction vector of known $\boldsymbol{f}^i$
6: $\quad \mathcal{E} \leftarrow \mathcal{E} \cup \boldsymbol{e}^i$
7: **end for**
8: $\boldsymbol{x}$-model $\leftarrow$ Train PS model $(\mathcal{E}, \mathcal{X})$

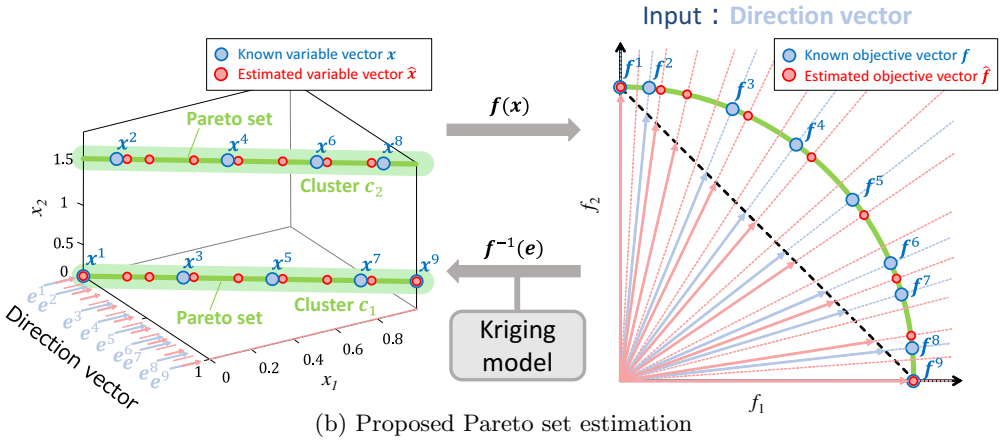
Estimation Process

9: $\mathcal{Q} \leftarrow \emptyset$ ▷ Solutions generated by PS estimation
10: **for each** $\hat{\boldsymbol{e}} \in \hat{\mathcal{E}}$ **do**
11: $\quad \hat{\boldsymbol{x}} \leftarrow \boldsymbol{x}$-model $(\hat{\boldsymbol{e}})$
12: $\quad \hat{\boldsymbol{f}} \leftarrow \boldsymbol{f}(\hat{\boldsymbol{x}})$
13: $\quad \mathcal{Q} \leftarrow \mathcal{Q} \cup (\hat{\boldsymbol{x}}, \hat{\boldsymbol{f}})$
14: **end for**
15: **return** $\mathcal{P}' \leftarrow$Extract non-dominated $(\mathcal{P} \cup \mathcal{Q})$

における位置 $\hat{\boldsymbol{x}}$(赤丸) を推定する処理に該当する．11 行目は，方向ベクトル $\hat{\boldsymbol{e}}$ を Kriging モデル $\boldsymbol{x}$-model に入力し，対応する推定変数ベクトル $\hat{\boldsymbol{x}}$ を得る．12 行目は，推定変数ベクトル $\hat{\boldsymbol{x}}$ を目的関数ベクトル $\boldsymbol{f}$ に入力し，目的関数値ベクトル $\hat{\boldsymbol{f}}$ を得る．13 行目は，新しい解 $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{f}})$ を解集合 $\mathcal{Q}$ に加える．10–14 行目の処理を，方向 $\hat{\boldsymbol{e}}$ を変えながら繰り返す．15 行目は，既知の解集合 $\mathcal{P}$ と生成した解集合 $\mathcal{Q}$ から非劣解集合 $\mathcal{P}'$ を抽出して出力する．



(a) Conventional Pareto set estimation [5)]



(b) Proposed Pareto set estimation

Fig. 2: Pareto set estimation of a multiodal problem

3.3 課題

従来の PS 推定は，マルチモダル問題において，推定精度が著しく低下するという課題がある．マルチモダル問題は，変数空間における複数の点が，目的空間の単一の点に写像される問題である．マルチモダル問題の一例を **Fig. 2 (a)** に示す．目的空間におけるパレートフロント上の各点が，変数空間の 2 点に対応している．より具体的には，目的空間における各方向ベクトル $\hat{\boldsymbol{e}}$ に対応する変数空間の点が，2 つある．このように，緑色の PS は 2 つに分かれている．従来の PS 推定は，目的空間における各方向ベクトル $\hat{\boldsymbol{e}}$ に対応する変数空間の位置 $\hat{\boldsymbol{x}}$ をひとつ求めるため，マルチモダルな PS を出力できない．結果として，**Fig. 2 (a)** に示されているように，赤丸で示された推定解は，緑色で示される真の PS から大きく乖離している．

4 提案法：マルチモダル PS 推定

4.1 概要

本稿では，マルチモダル問題における PS 推定の実現を目的とし，クラスタリングを用いた PS 推定法を提案する．提案法の概念図を **Fig. 2 (b)** に示す．提案法では，変数空間において既知の解集合 $\mathcal{P}$ をクラスタに分割する．**Fig. 2 (b)** では，既知の解集合 $\mathcal{P}$ が，クラスタ $C_1 = \{\boldsymbol{x}^1, \boldsymbol{x}^3, \boldsymbol{x}^5, \boldsymbol{x}^7, \boldsymbol{x}^9\}$ と $C_2 = \{\boldsymbol{x}^2, \boldsymbol{x}^4, \boldsymbol{x}^6, \boldsymbol{x}^8\}$ に分かれる．次に，クラスタごとに応答曲面を構築する．**Fig. 2 (b)** では，$\boldsymbol{x}_{C_1}$-model と $\boldsymbol{x}_{C_2}$-model を構築する．次に，双方の応答曲面に対して，一様に分布した膨大な方向ベクトル群 $\hat{\mathcal{E}}$ を入力し，それぞれから推定された赤い PS を得る．このように，異なる応答曲面にそれぞれの方向ベクトル $\hat{\boldsymbol{e}} \in \hat{\mathcal{E}}$ を入力することで，マルチモダル問題においても PS をより正確に推定できるようになる．

4.2 方法

提案法の疑似コードを **Alg. 2** に示す．従来法との差異は赤字で示している．8 行目では，既知の解集合を変数空間上でクラスタ群 $\mathcal{C} = \{C_1, C_2, \dots\}$ に分類する．本稿では，階層クラスタリング [8)] を活用した．階層クラスタリングの過程では，木構造のデンドログラムが作成される．提案法においては，変数空間における解の距離が近く，かつマルチモダルでないと判断されるクラスタの組み合わせから優先的に統合され，デンドログラムが構築される．なお，2 つのクラスタを統合した際に，変数値の変動が単調でないと判定される場合には，それらはマルチモダルであるとみなし，同一クラスタとして統合しない．**Fig. 2 (b)** の例では，このクラスタリング処理によって，$\mathcal{C} = \{C_1, C_2\}$ に分類される．9–13 行目は，各クラスタ $C_k \in \mathcal{C}$ について，PS 推定モデル $\boldsymbol{x}_{C_k}$-model を構築する．**Fig. 2 (b)** の例なら，$\boldsymbol{x}_{C_1}$-model と $\boldsymbol{x}_{C_2}$-model を構築する．14 行目は，大規模方向ベクトル群 $\hat{\mathcal{E}}$ について，PS 推定するための方向ベクトルに優先度を付ける．具体的には，**Alg. 3** において，1 行目は，大規模方向ベクトル群 $\hat{\mathcal{E}}$ の中から，要素に 1 を含む端の方向ベクトルを最優先に $\hat{\mathcal{E}}'$ として選ぶ．**Fig. 2 (b)** の例なら，$\hat{\mathcal{E}}' = \{\hat{\boldsymbol{e}}^1 = (0,1), \hat{\boldsymbol{e}}^{|\hat{\mathcal{E}}|=16} = (1,0)\}$ になる．2–5 行目は，すでに選択した $\hat{\mathcal{E}}'$ から最も遠い方向

Algorithm 2 Proposed Pareto Set Estimation

Require: Known solution set $\mathcal{P} = \{(\boldsymbol{x}^1, \boldsymbol{f}^1), (\boldsymbol{x}^2, \boldsymbol{f}^2), \dots, (\boldsymbol{x}^N, \boldsymbol{f}^N)\}$, a large unit vector set $\hat{\mathcal{E}} = \{\hat{\boldsymbol{e}}^1, \hat{\boldsymbol{e}}^2, \dots\}$
Ensure: Popoulation $\mathcal{P}'$

Learning Process
1: $\mathcal{F} = \{\boldsymbol{f}^1, \boldsymbol{f}^2, \dots, \boldsymbol{f}^N\} \leftarrow$ Extract objective vectors $(\mathcal{P})$
2: $\mathcal{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \dots, \boldsymbol{x}^N\} \leftarrow$ Extract variable vectors $(\mathcal{P})$
3: $\mathcal{E} \leftarrow \emptyset$ ▷ Direction vectors of known solutions
4: **for** $i \leftarrow 1, 2, \dots, N$ **do**
5: $\quad \boldsymbol{e}^i \leftarrow \boldsymbol{f}^i / \|\boldsymbol{f}^i\|_1$ ▷ Direction vector of known $\boldsymbol{f}^i$
6: $\quad \mathcal{E} \leftarrow \mathcal{E} \cup \boldsymbol{e}^i$
7: **end for**
8: $\mathcal{C} = \{C_1, C_2, \dots\} \leftarrow$ Clustering $(\mathcal{E}, \mathcal{X})$
9: **for each** $C_k \in \mathcal{C}$ **do**
10: $\quad \mathcal{E}_{C_k} \leftarrow$ Extract direction vectors (C_k)
11: $\quad \mathcal{X}_{C_k} \leftarrow$ Extract variable vectors (C_k)
12: $\quad \boldsymbol{x}_{C_k}$-model $\leftarrow$ Train PS model $(\mathcal{E}_{C_k}, \mathcal{X}_{C_k})$
13: **end for**
14: $\hat{\mathcal{E}}' \leftarrow$ Prioritize Directions $(\hat{\mathcal{E}})$ ▷ **Algorithm 3**

PS Estimation
15: $\mathcal{Q} \leftarrow \emptyset$ ▷ Solutions generated by PS estimation
16: **for each** $\hat{\boldsymbol{e}} \in \hat{\mathcal{E}}'$ **do**
17: $\quad \mathcal{C}' \leftarrow$ Select Target Clusters $(\hat{\boldsymbol{e}}, \mathcal{C})$ ▷ **Algorithm 4**
18: $\quad$ **for each** $C_k \in \mathcal{C}'$ **do**
19: $\quad\quad \hat{\boldsymbol{x}} \leftarrow \boldsymbol{x}_{C_k}$-model $(\hat{\boldsymbol{e}})$
20: $\quad\quad \hat{\boldsymbol{f}} \leftarrow$Evaluation$(\hat{\boldsymbol{x}})$
21: $\quad\quad \mathcal{Q} \leftarrow \mathcal{Q} \cup (\hat{\boldsymbol{x}}, \hat{\boldsymbol{f}})$
22: $\quad\quad$ **if** $|\mathcal{Q}| \geq |\hat{\mathcal{E}}|$ **then**
23: $\quad\quad\quad$ **return** $\mathcal{P}' \leftarrow$Extract non-dominated $(\mathcal{P} \cup \mathcal{Q})$
24: $\quad\quad$ **end if**
25: $\quad$ **end for**
26: **end for**

Algorithm 3 Prioritize Directions

Require: A large unit vector set $\hat{\mathcal{E}} = \{\hat{\boldsymbol{e}}^1, \hat{\boldsymbol{e}}^2, \dots, \hat{\boldsymbol{e}}^{|\hat{\mathcal{E}}|}\}$
Ensure: Prioritized unit vector set $\hat{\mathcal{E}}'$
1: $\hat{\mathcal{E}}' \leftarrow$ Pick extreme vectors with a element of 1 from $\hat{\mathcal{E}}$
2: **while** $|\hat{\mathcal{E}}'| < |\hat{\mathcal{E}}|$ **do**
3: $\quad \hat{\boldsymbol{e}} \leftarrow$ Pick the farthest vector from $\hat{\mathcal{E}}$
4: $\quad \hat{\mathcal{E}}' \leftarrow \hat{\mathcal{E}}' \cup \hat{\boldsymbol{e}}$
5: **end while**
6: **return** $\hat{\mathcal{E}}'$

ベクトル $\hat{\boldsymbol{e}}$ から優先的に $\hat{\boldsymbol{e}}$ を $\hat{\mathcal{E}}'$ に繰り返し追加する．**Fig. 2 (b)** の例なら，$\hat{\mathcal{E}}' = \{\hat{\boldsymbol{e}}^1, \hat{\boldsymbol{e}}^{|\hat{\mathcal{E}}|=16}, \hat{\boldsymbol{e}}^8, \hat{\boldsymbol{e}}^{12}, \dots\}$ という具合になる．これにより，大規模方向ベクトル群 $\hat{\mathcal{E}}$ を網羅できなかったとしても，目的空間において均一に分布する方向ベクトルを対象に，PS 推定を実行する．**Alg. 2** に戻ると，16–26 行目において，優先度付けした大規模方向ベクトル群 $\hat{\mathcal{E}}'$ の順番に各方向ベクトル $\hat{\boldsymbol{e}} \in \hat{\mathcal{E}}'$ に注目しながら PS 推定を実行する．17 行目は，注目する方向ベクトル $\hat{\boldsymbol{e}}$ に対応するクラスタ群 $\mathcal{C}'$ を取り出す．具体的には，**Alg. 4** において，各クラスタ C_k の既知の解集合の方向ベクトルの範囲に注目する方向ベクトル $\hat{\boldsymbol{e}}$ が含まれる場合は，該当するクラスタ C_k を $\mathcal{C}'$ に追加する処理を繰り返す．**Alg. 2** に戻ると，18–25 行目は，取り出したクラスタ群 $\mathcal{C}'$ の各クラスタ $C_k \in \mathcal{C}'$ の応答局面 $\boldsymbol{x}_{C_k}$-model から，注目する方向ベクトル $\hat{\boldsymbol{e}}$ に対応する変数ベクトル $\hat{\boldsymbol{x}}$ を求める．つまり，注目するひとつの方向ベクトル $\hat{\boldsymbol{e}}$ に対して，$|\mathcal{C}'|$ 個の変数ベクトルを生成する．22 行目では，$|\hat{\mathcal{E}}|$ 個の変数ベクトルを生成した時点で，アルゴリズムを終了する．つまり，大規模方向ベクトル群 $\hat{\mathcal{E}}$ を網羅しない場合がある．そのため，14 行目において，目的空間における均一性の高い方向ベクトルの順番を求めている．

Algorithm 4 Select Target Clusters

Require: A direction vector $\hat{\boldsymbol{e}}$, Cluster set $\mathcal{C} = \{C_1, C_2, \dots\}$
Ensure: Selected clusters $\mathcal{C}'$
1: $\mathcal{C}' \leftarrow \emptyset$
2: **for each** $C_k \in \mathcal{C}$ **do**
3: $\quad$ **if** Direction range of cluster C_k involves $\hat{\boldsymbol{e}}$ **then**
4: $\quad\quad \mathcal{C}' \leftarrow \mathcal{C}' \cup C_k$
5: $\quad$ **end if**
6: **end for**
7: **return** $\mathcal{C}'$

4.3 期待される効果

提案法は，変数空間上で既知の解集合 $\mathcal{P}$ をクラスタリングすることにより，マルチモダリティを識別する．各クラスタ C_k ごとに $\boldsymbol{x}_{C_k}$-model を構築することで，変数空間における局所的な応答曲面を得る．さらに，各 $\boldsymbol{x}_{C_k}$-model に方向ベクトル $\hat{\boldsymbol{e}}$ を入力して局所的に PS 推定を行うことで，マルチモダルな変数空間においても PS 推定の精度向上が期待できる．

5 実験設定

5.1 方法

従来法 [5] と提案法を比較する．大規模方向ベクトル群 $\hat{\mathcal{E}}$ のサイズは，$|\hat{\mathcal{E}}| = 1000$ とした．

5.2 問題

マルチモダルな多目的テスト問題として，$M = 2$ 目的 $D = 2$ 変数の MMF1–8 [6]，および $M = 2$ 目的 $D = 30$ 変数の LIRCMOP1-2 [7] を用いた．既知解集合 $\mathcal{P}$ は，進化計算によって十分に最適化した結果を用いた．各問題における既知解集合 $\mathcal{P}$ のサイズ（$|\mathcal{P}| = N$）を **Table 1** に示す．

5.3 評価

本稿では，評価指標として Hypervolume（HV）[9] を用いる．HV は，得られた解集合 $\mathcal{P}$ と参照点 $\boldsymbol{r}$ によって囲まれる目的空間上の M 次元体積である．本稿では，$M = 2$ の目的問題のみを扱うため，HV は 2 次元の面積となる．HV は，真のパレートフロントに対する収束性と多様性が高い解集合 $\mathcal{P}$ ほど大きくなる．さらに，$\mathcal{P}$ のサイズが大きい場合にも，HV は高くなる傾向がある．各問題に対して得られたすべての解集合において，目的関数値を [0,1] に正規化し，参照点 $\boldsymbol{r} = (1.1, 1.1)$ を用いて HV を算出した．

6 実験結果と考察

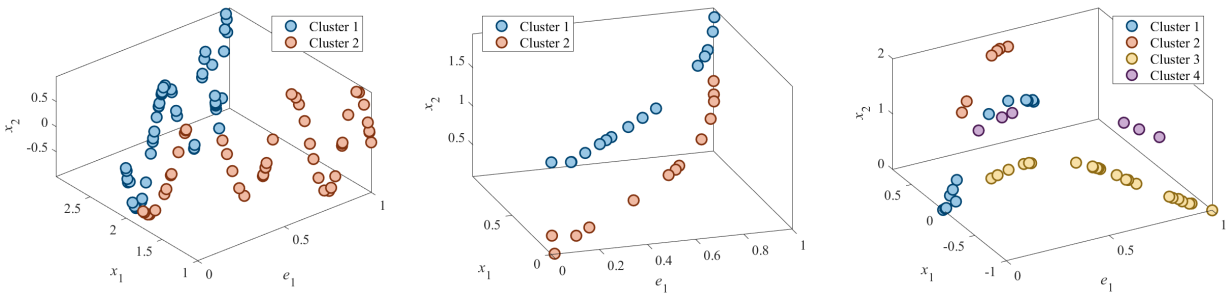
6.1 変数空間における解集合クラスタ

MMF1，MMF2，MMF4 問題に注目し，既知の解集合 $\mathcal{P}$ の変数空間における分布および，提案法によって得られたクラスタリング結果を **Fig. 3** に示す．なお，**Fig. 1** および **2** の左図と同様に，変数空間に方向ベクトルの要素 e_1 の軸を加えて表示している．

これらの結果から，既知の解集合 $\mathcal{P}$ は，MMF1 および MMF2 では 2 クラスタ，MMF4 では 4 クラスタに分類されることがわかる．

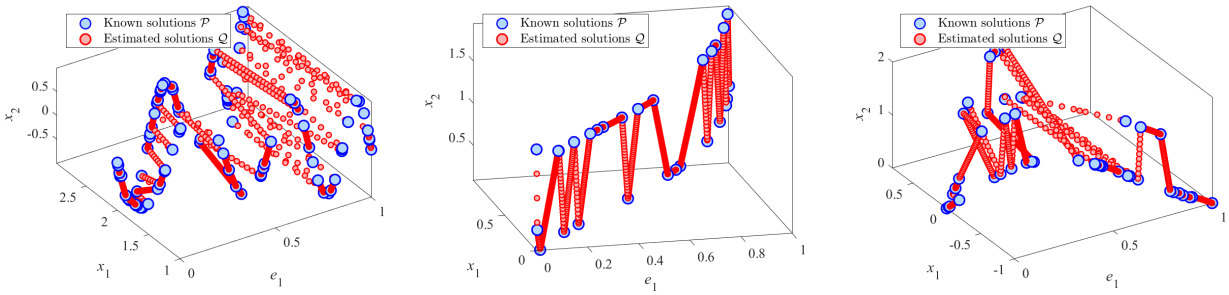
6.2 変数空間における解分布

MMF1，MMF2，MMF4 問題における，得られた解集合 $\mathcal{P}$ の変数空間での分布を観察する．**Fig. 4** に従来法，**Fig. 5** に提案法の結果を示す．図中の青点は既知の解，赤点は PS 推定によって得られた解を示す．



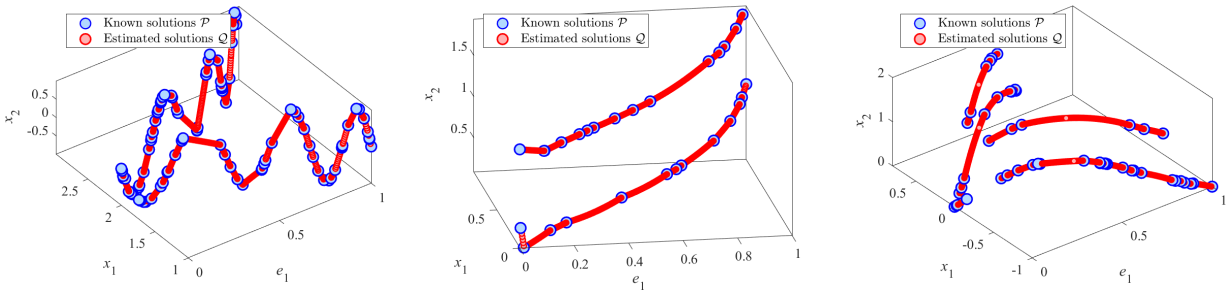
(a) MMF1 (b) MMF2 (c) MMF4

Fig. 3: Clusters obtained by the proposed method



(a) MMF1 (b) MMF2 (c) MMF4

Fig. 4: Known solutions and estimated ones obtained by the conventional method [5)]



(a) MMF1 (b) MMF2 (c) MMF4

Fig. 5: Known solutions and estimated ones obtained by the proposed method

Fig. 4 に示すように，従来法による推定解（赤点）は，複数の既知解（青点）のクラスタの外側に多く分布していることがわかる．一方，**Fig. 5** に示すように，提案法による推定解（赤点）は，各既知解クラスタ（青点）の内部に分布していることが確認できる．

6.3 目的空間における解集合

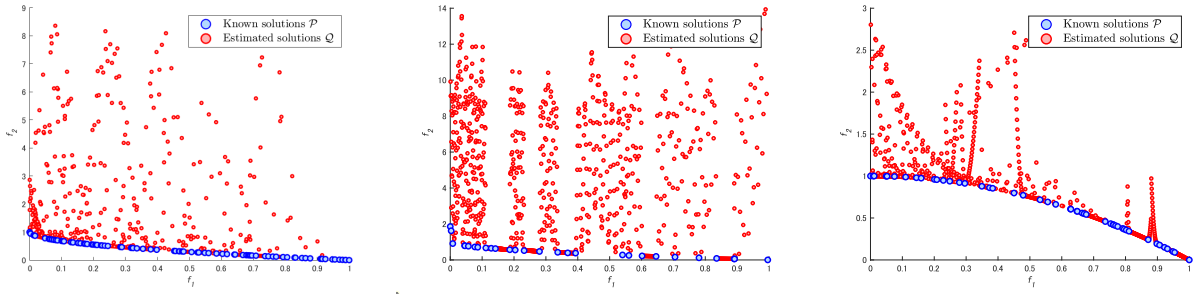
得られた解集合 $\mathcal{P}$ の目的空間における分布について議論する．

Fig. 6 および **Fig. 7** は，従来法および提案法によって得られた解集合の分布を示す．**Fig. 6** に示す従来法の結果からは，劣解が多数生成されていることがわかる．これは，**Fig. 4** に示すように，従来法が既知の解集合のクラスタの外側に解を生成していることに起因する．一方，**Fig. 7** に示す提案法の結果からは，劣解の生成が抑制されていることが確認できる．これは，**Fig. 5** に示すように，提案法が既知の解集合のクラスタ内に解を生成しているためである．

従来法および提案法により得られた解集合 $\mathcal{P}$ に基づく HV の値を **Table 1** に示す．ここでは，両手法のうち HV が高い方を太字で示している．これらの結果から，提案法は多くの問題において従来法を上回る HV を達成していることがわかる．ただし，LIRCMOP1 においては，マルチモダル性が複雑でクラスタ数が多くなった影響により，提案法の実行が困難であった．また，MMF6 のようにマルチモダル性が極めて複雑な問題では，提案法は従来法よりも低い HV を示した．一方，LIRCMOP2 のように変数次元が高い問題においても，提案法は従来法より高い HV を達成できることが確認された．

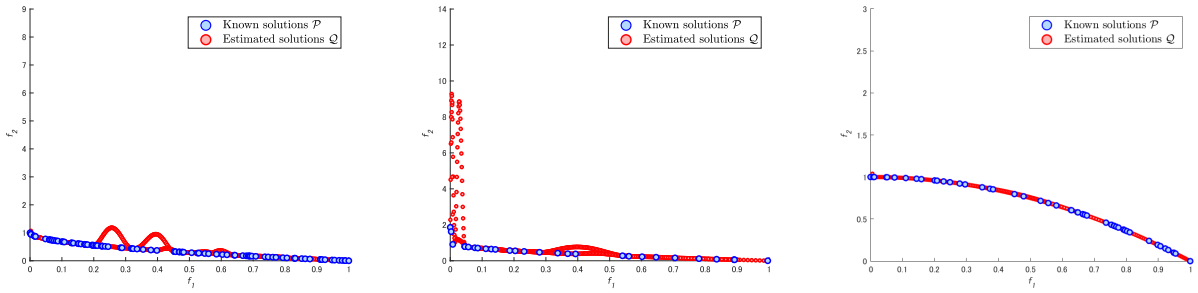
7 まとめ

本稿では，マルチモダル問題における PS 推定の精度向上を目的として，クラスタリングを用いた PS 推定法を提案した．提案法は，変数空間上で既知の解集合をクラスタリングし，各クラスタごとに独立に PS を推定する．マルチモダルなテスト問題である MMF1–8 および LIRCMOP1–2 を用いた実験の結果，提案法は，マルチモダル性が高い問題を除き，劣解の生成を抑制できることを明らかにした．これは，変数空間において



(a) MMF1 (b) MMF2 (c) MMF4

Fig. 6: Known solutions and estimated ones obtained by the conventional method [5)]



(a) MMF1 (b) MMF2 (c) MMF4

Fig. 7: Known solutions and estimated ones obtained by the proposed method

Table 1: HV values of the obtained solutions $\mathcal{P}'$

	MMF1	MMF2	MMF3	MMF4	MMF5	MMF6	MMF7	MMF8	LIRCMOP1	LIRCMOP2
Number of variables D	2	2	2	2	2	2	2	2	30	30
Number of known solutions N	98	28	31	48	40	98	48	48	40	40
Conventional Method	0.7214	0.7037	0.7060	0.4455	0.7096	**0.7202**	0.7229	0.3488	**0.2269**	0.3608
Proposed Method	**0.7225**	**0.7140**	**0.7201**	**0.4481**	**0.7121**	0.7196	**0.7232**	**0.3502**	–	**0.3627**

既知の解集合を分布に基づいてクラスタに分類し，各クラスタ内に解を生成することによって達成された．

今後は，より複雑なマルチモダル性を有する問題や，次元数の多い問題において，解の推定精度を高める手法の構築に取り組む．

参考文献

1) K. Deb：Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, (2001)

2) C. A. C. Coello, G. B. Lamont, D. A. Van Veldhuizen：Evolutionary Algorithms for Solving Multi-Objective Problems, Springer, (2007)

3) J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn：Design and Analysis of Computer Experiments, Statistical Science, **4**-4, 409/423, (1989)

4) M. L. Stein：Interpolation of Spatial Data: Some Theory for Kriging, Springer Science & Business Media, (2012)

5) T. Takagi, K. Takadama, and H. Sato: Supervised Multi-objective Optimization Algorithm Using Estimation, 2022 IEEE World Congress on Computational Intelligence (WCCI 2022), 1/8, (2022)

6) C. Yue, B. Qu, and J. Liang：A Multiobjective Particle Swarm Optimizer Using Ring Topology for Solving Multimodal Multiobjective Problems, IEEE Transactions on Evolutionary Computation, **22**-5, 805/817, (2018)

7) Z. Fan, W. Li, X. Cai, H. Huang, Y. Fang, Y. You, J. Mo, C. Wei, and E. Goodman：An improved epsilon constraint-handling method in MOEA/D for CMOPs with large infeasible regions, Soft Computing, **23**, 12491/12510, (2019)

8) X. Ran, Y. Xi, Y. Lu, X. Wang, and Z. Lu: Comprehensive Survey on Hierarchical Clustering Algorithms and The Recent Developments, Artificial Intelligence Review, **56**, 8219/8264, (2023)

9) E. Zitzler, L. Thiele：Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach, IEEE Transactions on Evolutionary Computation, **3**-4, 257/271, (1999)