CH-003

深層学習による三次元折り紙形状推定における視点位置の影響
Effect of Viewpoint Position on 3D Origami Shape Estimation Using Deep Learning

中垣内 千晶† 西野 隆典†† 武田一哉†††
Chiaki Nakagaito Takanori Nishino Kazuya Takeda

1. はじめに

日常生活で扱うことがある様々な物体を，ロボットハンドなどによって自動的に操作できるようにすることが，重要なタスクの一つになっている．ロボットハンドで物体を把持するためには，物体の三次元形状があらかじめわかっていることが望ましいが，単一のカメラやセンサでは死角が生じるため，形状を完全に知ることは困難である．また，さらに高度な技術として，ロボットハンドなどによって折り紙のような，折りたたみ構造をもつ物体を自動的に折りたたむ場合には，その三次元形状を推定できることが望ましい．三次元形状を正確に推定できるようになれば，より適切に折り畳み操作を行うことが可能になると考えられる．

折り紙のような三次元の折りたたみ構造を持つ物体の三次元形状を推定するためには，物体外部の単一視点位置からは見えないオクルージョン領域がある場合にも，適切に推定でき，マーカーレスでも推定できることが望ましい．折り紙の形状推定や，ロボットハンドで折り紙を折る研究がこれまでにある[1-3] 一方で，マーカーレスで，三次元点群情報のみから，オクルージョンのある折り紙の三次元形状を推定することが課題となっている．この課題を解決するには，探索による方法などが考えられるが，この方法では，推定時の計算量の問題や，オクルージョンに対する問題などがある．たとえば，単純な全探索を適用する場合は，点群数と展開図の辺および頂点の数によってはリアルタイムに推定することが難しいと考えられる．そのため本稿では，形状推定実行時にあまり時間がかからず，オクルージョンにもロバストに動作する可能性があるなどの利点から，深層学習を用いて推定を行う．

本稿では，ある決まった展開図に限定し，マーカーレスで，オクルージョンのある折り紙の三次元点群データのみから三次元形状を推定する方法を提案する．また，視点位置を変化させて，視点位置によって推定にどのような影響があるか調査した．折り紙の三次元形状については，シミュレーションにより実際の折り紙形状を模擬することが可能と考えられるため，折り紙を折る過程をシミュレーションすることで複数視点位置の三次元点群データを作成し，深層学習によって三次元形状を推定した．また，実験では，視点位置によって推定精度にどのような影響があるかを調査した．

本稿では，2 節で関連研究を紹介し，3 節で提案手法および構築したデータセットについて説明し，4 節で各視点位置の推定精度を調査する実験について説明する．5 節で考察と課題を述べ，6 節でまとめを述べる．

†名古屋大学 大学院情報学研究科 Graduate School of Informatics, Nagoya University
††名城大学都市情報学部 Faculty of Urban Science, Meijo University
†††名古屋大学未来社会創造機構 Institutes of Innovation for Future Society, Nagoya University

2. 関連研究

2.1 折り紙の形状推定

折り紙の形状推定や，折り紙を折るシミュレーション，ロボットによる折り紙操作などの研究はこれまでにも多く提案されている[1-3].

折り紙の形状推定に関する研究には，二次元バーコードを用いることで折りたたみ構造を容易に認識できるようにした研究[1] がある．この研究では平坦に折りたたまれるケースのみを対象としているが，特定の展開図に限定せず，さまざまな折った状態を認識することができる．

ロボットハンドによって折り紙を折る研究には，文献[2] や[3] などがある．これらの研究では，紙にマーカーを付けて，物理シミュレーションなどを用いて三次元形状を推定している．

また，折り紙を折るシミュレーションに関する研究では，さまざまな方法がこれまでに提案されており，折り紙を折る過程のアニメーションや三次元メッシュデータを作成できるソフトウェアも多く公開されている[4-6].

2.2 三次元形状を扱う深層学習

三次元形状の深層学習に関する研究はこれまでに多くあり，特に三次元点群を扱う場合には，voxel-based の方法と，point-based の方法に大別される．

voxel-based の方法では，三次元点群を voxel 化して深層学習を適用する．この方法の利点として，データ構造が単純であることなどが挙げられるが，この方法の難点として，メモリコストの問題，解像度が限定的であることなどがある．これらの課題を改善するために，Octree を用いる方法があり，Octree-CNN[7]や Adaptive Octree-CNN[8] などが提案されている．

一方，point-based の方法の利点には，他のデータ形式に変換する際のオーバーヘッドを削減できることや，変換にともなう情報の損失を回避できることがある．point-based の方法で三次元点群に深層学習を適用することは，点群の順序普遍性，局所的な特徴，回転不変の課題のために困難だったが，これらに対してロバストな方法が，PointNet[9] によって提案された．三次元点群を深層学習によって再構成する研究はこれまでにも多くあり[9-15]，AtlasNet[10], FoldingNet[11] などが提案されている．

深層学習を用いて，椅子や車などの三次元形状を推定する研究や，バーコードやマーカーを使用した折り紙の形状推定の研究がこれまでにある一方で，マーカーなどを使わずにオクルージョンのある折り紙の三次元形状を推定する研究や，視点位置による推定精度への影響を調査した研究は少ない．

3. 提案手法

提案する深層学習ネットワークと，実験で用いる三次元点群データを作成する方法について述べる．

3.1 折り紙のための深層学習ネットワーク

本研究では，紙を折った状態を推定するために，PointNet[9]や AtlasNet[10]などの既存の三次元点群再構成ネットワークをもとに，ネットワークを構築した．提案手法の概要を図１に示し，ネットワーク構成を図２に示す．ネットワークの構成は，Point Net[9]および AtlasNet[10]をもとにしている．Encoder 部分は PointNet[9] の Encoder ベースの構成になっている．PointNet[9] の Encoder は，点群の順序や回転に対してロバストであることが利点である．また，AtlasNet[10]の，ベースとなる二次元グリッドからそれを変形した三次元メッシュを出力する仕組みを用いて，折り紙の展開図を変形した三次元メッシュデータを出力するように変更して実装した．提案手法では，ある一つの展開図をベースに，展開図で折った三次元形状を出力することができ，ベースの展開図には任意のパターンを使用することができる．ネットワークの入力は三次元点群で，出力は，展開図を折った形の三次元メッシュになっている．損失関数には，MSE Loss を使用した．Encoder 出力の中間層ベクトルは，予備実験の結果から，32 次元に設定した．

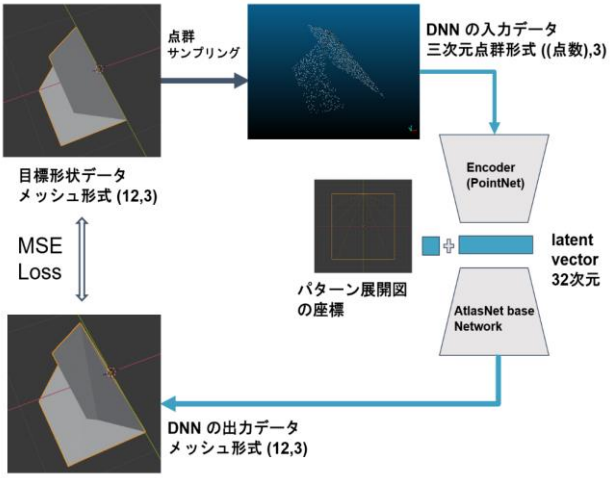

図１：提案手法の構成

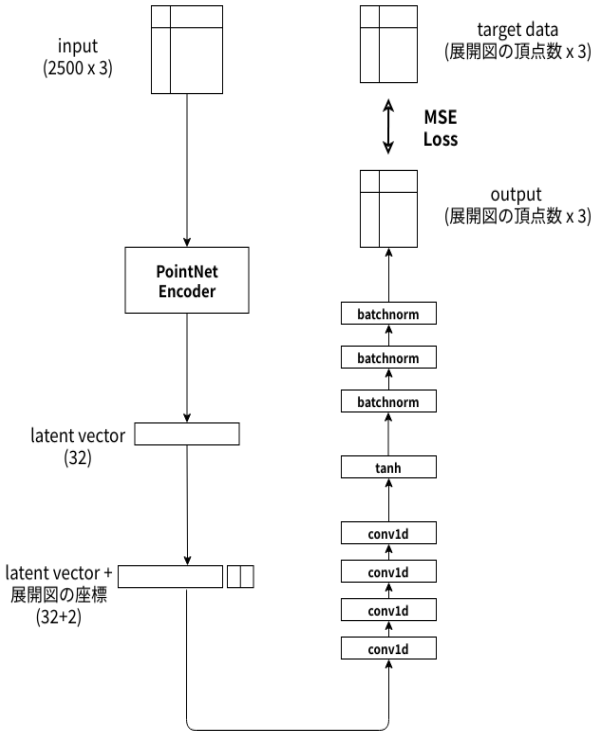

図 ２： ネットワーク構成

3.2 データセットの構築

深層学習に使用できる既存の三次元点群データセットには，ShapeNet Dataset[18]，ModelNet Dataset[19] などがこれまでに公開されている．これらのデータセットは，机，椅子，飛行機，車などの複数のカテゴリからなる三次元点群データセットである．一方で，折り紙の三次元点群のデータセットはあまり公開されていない．既存の折り紙のシミュレーションソフトウェアでは，折り紙を折る過程のアニメーションや，三次元メッシュデータを作成することができる[4-6]．しかし，深層学習に必要な，折り紙の折り過程の三次元点群の利用可能なデータセットはこれまでに公開されていない．そこで本研究では，シミュレーションによって新たな折り紙の三次元点群データセットを構築した．作成したデータセットの概要と，作成手順を以下に述べる．

3.2.1 データセットの概要

実験で用いたデータセットの概要を述べる．

- 折り紙パターン：紙飛行機パターン
- 折る過程の状態数：1,440
- 視点位置数：62
- 点群サンプリング方法：
  1) メッシュ上の全ての領域からランダムサンプリングする方法
  2) 単一視点位置から見える領域のみからサンプリングする方法
- サンプリング点数：サンプリング方法 1 では 3,000 点，サンプリング方法 2 では 0 から 3,000 点
- 三次元点群の位置座標範囲：$-1 < x, y, z < 1$
- 初期正方形の一辺の長さ：1.6
- データ形式：メッシュデータおよび三次元点群 (.ply)

視点位置には，原点を中心とした球面の半径を 3 に設定し，球面上に 30 度刻みで視点位置を合計 62 個設定した．球面上の視点位置座標は，以下のように計算した値を用いた．

$$\begin{cases} x = r\sin\theta\cos\phi \\ y = r\sin\theta\sin\phi \\ z = r\cos\theta \end{cases} \quad \begin{pmatrix} r = 3 \\ 0° \le \theta \le 180° \\ 0° \le \phi < 360° \end{pmatrix} \tag{1}$$

ここで，rは球面の半径，θ よびϕ は角度で，それぞれ鉛直角，水平角を表す．

データの種類は以下の 3 種類に分類される．
Data 1:　三次元メッシュデータ．紙飛行機パターンを折る過程 1,440 状態
Data 2:　Data 1 のメッシュデータをもとに，メッシュ上の全ての領域から三次元点群をランダムサンプリングしたデータ
Data 3:　Data 2 のデータをもとに，視点位置から見える領域のみから三次元点群をサンプリングしたデータ

以降では，上記のデータをこの Data 1，Data 2，Data 3 の番号を使って説明する．図 3 にメッシュデータのレンダリング画像の例を示す．また，Data 1 から Data 3 のデータの例をそれぞれ図 4 から図 6 に示す．これらの図では，x 軸を赤，y 軸を緑，z 軸を青で表示している．また，矢印の向きに正の値をとる．

なお，全ての領域からサンプリングした Data 2 では，それをもとに作成するデータ Data 3 と一部で三次元点群が同一の位置になる事を防ぐために，点群の分布位置が異なる二つのデータを作成した．

3.2.2 データセット作成手順

構築したデータセットの作成手順を述べる．

まず，Data 1 のデータの作成手順は以下の通りになっている．

1. 平坦折り展開図パターンを既存のソフトウェアである ORIPA[20] で作成する．
2. 1.で作成したデータを読み込み，各頂点位置が-1.0 から 1.0 の範囲に収まるように設定する．
3. 折りパターンの折り目で面を回転させる方法で，シミュレーションを実行する．0.5 度刻みで 1,440 個の ply 形式のデータを出力する．図 4 に出力データ例を示す．

次に，Data 2 のデータの作成手順を述べる．

1. Data 1 の各データで，3,000 点をメッシュ面上の全ての領域からランダムサンプリングし，データを出力する．
2. 1. と同様の方法で，点群の分布位置が異なるデータを出力する．図 5 に出力データ例を示す．

次に，Data 3 のデータの作成手順を述べる．

1. Data 1 のメッシュデータを用意する．
2. 1.　に対応する Data 2 のデータを用意する．なお，Data 2 のデータの分布位置 2 パターンのうち片方の 1 パターンのデータを使用する．
3. 視点位置座標として，球面上の位置座標を計算する．
4. 3.で設定した視点位置から，2. の点群の各点の方向の ray を計算する．
5. 3. で計算した ray が 1. のメッシュデータにヒットした場所のうち，視点位置から最も近い点と，対応するサンプリング点との間のユークリッド距離を計算する．
6. 5. で計算した距離が 0 に近い場合は，対応するサンプリング点を残し，そうでなければ残さずに消去する．0 に近いかどうかは，閾値にごく小さい定数と

図 3 ：紙飛行機パターンメッシュデータのレンダリング画像の例

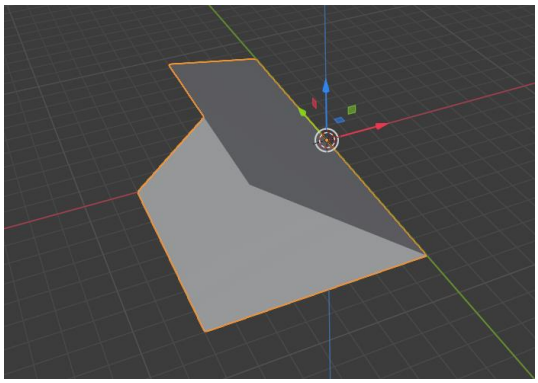

図 4 ：Data 1 三次元メッシュデータの例

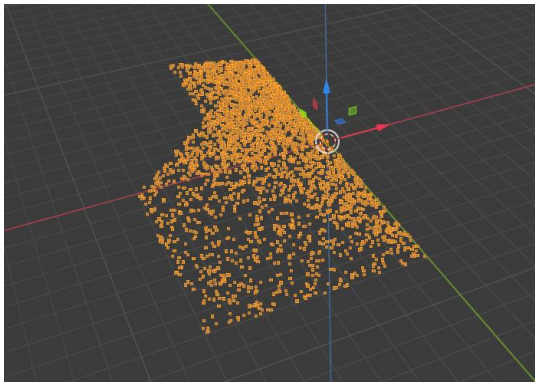

図 5 ：Data 2 全ての領域からランダムサンプリングしたデータの例

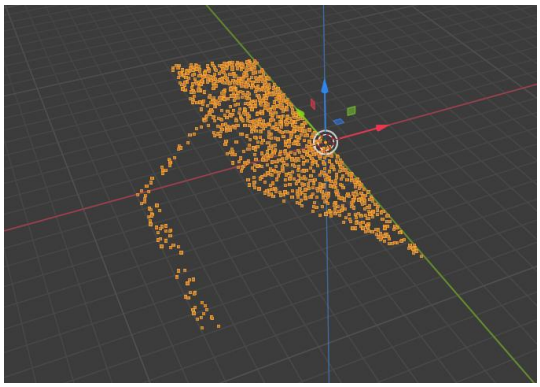

図 6 ：Data 3 単一視点位置から見える領域のみからサンプリングしたデータの例

して10^{-7} を設定し，その閾値によって判定している．

7. 全てのサンプリング点について上記の方法で計算し，残った点群データを出力する．図 6 に出力データ例を示す．

このデータの注意点としては，複数枚分が完全に折り重なっている領域で，Data 2 のデータではその枚数分サンプリングしている一方，Data 3 のデータでは，枚数分ではなく一枚分のみサンプリングしていることが挙げられる．

なお，これらのデータセットを GitHub にて公開した．

URL: https://github.com/0010cha/data_origami

4. 実験

4.1 実験条件

推定形状が目標形状に近い出力ができているかを，62 個の視点位置ごとに評価する．学習およびテストに使用したデータを以下に示す．

- 学習入力データ：Data 2 のデータ 1,440 ファイル
- テストデータ： Data 3 のデータ　一つの視点位置あたり 1,440 ファイル，各視点位置 62 点についてそれぞれ評価
- 目標データ：Data 1 のデータ 1,440 ファイル

続いて，ネットワークの学習条件を以下に示す．

・Batch size: 10
・optimizer: Adam optimizer (learning rate = 0.05)
・epoch 数: 30
・Loss function: MSE Loss
・Pytorch

Pytorch で deterministic に設定し，学習結果が再現されるようにした．

4.2 実験結果

Test Loss の平均を，視点位置ごとに，(Test Loss の総和) / (使用データ数) で計算した．ただし，$\theta = 90°$ の場合（$z = 0$ の場合に相当する）に三次元点群数が 0 になる場合のデータは除いて計算している．データ点数の平均値を図 7 に，各視点位置での Test Loss の平均の値を図 8 に示す．

実験結果では，$\theta = 90°$ の場合に，その他の角度の場合と比べて，Test Loss の平均が大きい結果となった．また，$\theta < 90°$ の場合（視点位置が $z > 0$ にある場合に相当する）は，$\theta > 90°$ にある場合（$z < 0$ の場合に相当する）と比べて，比較的 Test Loss 平均が小さい傾向が見られた．($\theta = 30°$，$\phi = 150°$) の場合($z > 0$ かつ，真上より少し斜めの位置に視点位置がある場合) に，Test Loss 平均が最小となった．参考として，この場合の Training Loss と Test Loss の推移グラフを図 9 に示す．また，($\theta = 90°$，$\phi = 90°$) の場合 ($z = 0$ の場合) に Test Loss の平均が最大になった．

また，単一視点位置から見える領域からのみサンプリングしているデータを使用しているため，視点位置および形状によって三次元点群のサンプリング点数が異なる．各視点位置で，データの三次元点数の平均を計算した．この結果を図 7 に示す．結果から，$\theta = 90°$の場合は，その他の角度の場合と比べて，比較的三次元点群点数の平均が少な

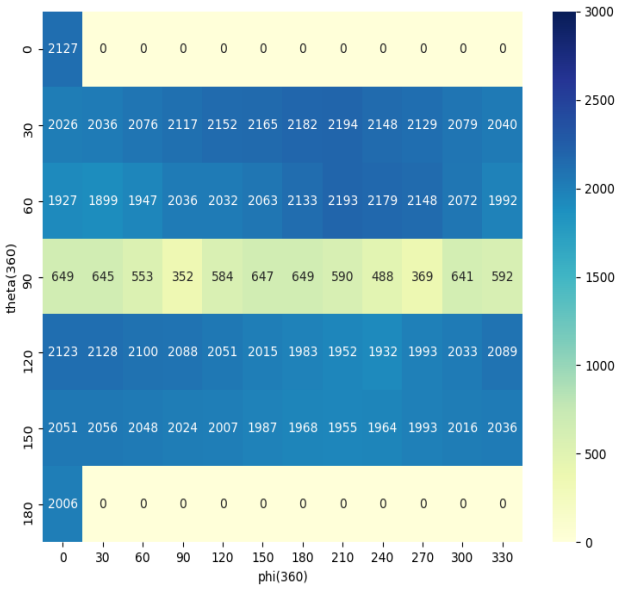

図 7 ：視点位置ごとのデータ点数平均

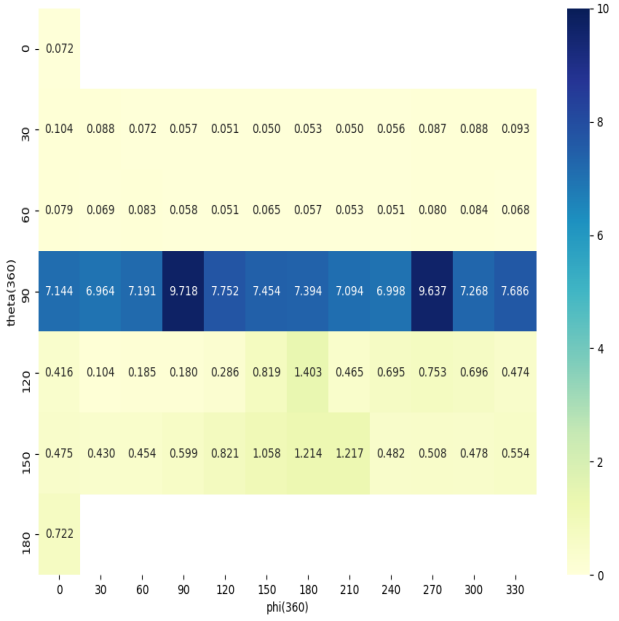

図 8 ：視点位置ごとの Test Loss (*100)

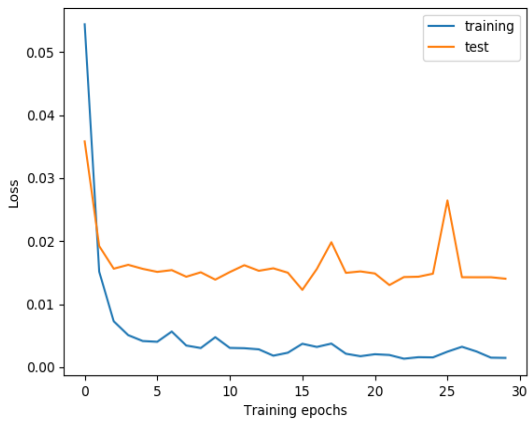

図 9 ：Training Loss と Test Loss の推移

い傾向になった．Test Loss の平均は最小値が 0.000501 ($\theta = 30°$，$\phi = 150°$) ，最大値は 0.097178 ($\theta = 90°$，$\phi = 90°$)であった．なお，xy 平面上以外の視点位置では，($\theta = 120°$，$\phi = 180°$)のときに 0.014033 となり最大となった．

5. 実験結果の分析

5.1 考察

実験結果によって得られる観察と考察を述べる．

実験結果から，紙飛行機パターンに限定しているが，学習データと異なる，オクルージョンがあるデータに対しても，視点位置によってはおよそ三次元形状の推定が可能であると考えられる．参考として，Test Loss の平均 が最小になった視点位置の場合の入力データ例を図１０に示し，対応する出力結果例を図１１に示す．また，Test Loss の平均が最大になった視点位置の場合の失敗例として，目標データを図１２に，対応する入力データを図１３に，出力結果を図１４に示す．視点位置が，初期状態の正方形と同一の平面上にあるような場合は比較的推定精度が悪く，正確に推定できない場合があることが確認できた．また，各視点位置での結果の比較から，視点位置によって推定精度が異なることを確認した．

また，サンプリング点数平均の多さと Test Loss の小ささには，ある程度関連があるように考えられるが，Test Loss の平均が最小になった位置や $\theta = 90°$ 以外の場合で Test Loss 平均が最大になった位置などと，サンプリング点数平均が最大の位置および $\theta = 90°$ 以外で最小の位置は，一致しない結果になった．これは，視点位置から見えないオクルージョン領域の変化に伴い，データのサンプリング点数も変化することで，ある程度の関連が出ていると考えられる．また，点数の変化のみではなく，全体の形状を比較的とらえやすい位置にあるかなどの他の要因も影響している可能性があると考えられる．

さらに，視点位置 $z > 0$ にある場合のほうが， $z < 0$にある場合に比べて，Test Loss 平均が小さい傾向が見られたが，この理由には，折る手順５つのうち４つが，$z > 0$で変形しており変形過程の形状を比較的とらえやすかったためと考えられる．

5.2 課題

実験結果，および考察を踏まえて提案手法の課題を述べる．

一つ目は，提案手法はネットワークの出力が現実に折ることが可能な状態であることを保証していない．出力メッシュが自己交差を含む場合や，出力メッシュの各辺の長さがもとの展開図と厳密に同じ長さではない．これらの問題点は，ネットワークの損失関数に適切な正則化項を設定することである程度改善する可能性がある．

二つ目は，実世界の点群センサーデータを使用していないことが挙げられる．今回用いたシミュレーションデータが，実世界の点群センサーデータと同一であるとは限らない．そのため，シミュレーションデータだけでなく，実世界の点群センサーデータを用いた実験をする必要がある．

三つ目は，学習に使用するデータ数の課題がある．今回の実験では，紙飛行機パターンのみを使用しており，学習用に使用できるデータ数が少ない条件であった．今後，より多くの折り方パターンを選定し，データを作成する必要がある．

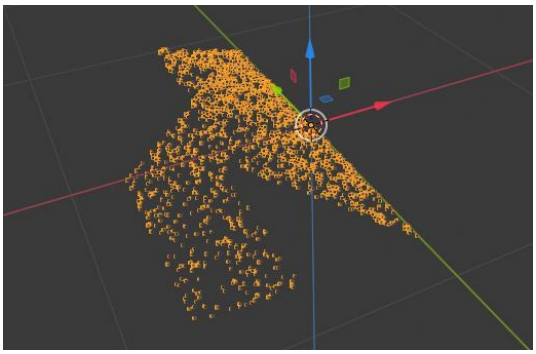

図 １０：入力データ例 (Test Loss が最小の場合)

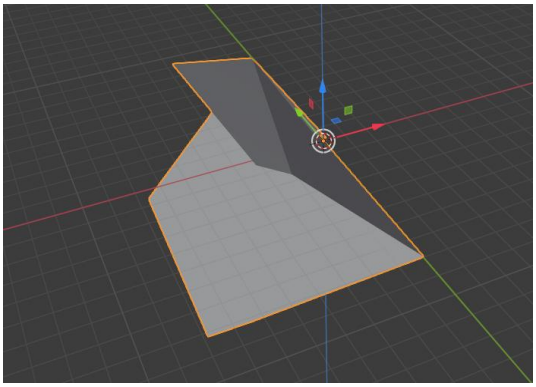

図 １１：ネットワーク出力結果例 (Test Loss が最小の場合)

6. まとめ

折り紙の三次元点群から，ある展開図で折った三次元形状を，深層学習を用いてマーカーレスで推定する方法を提案した．また，提案する深層学習ネットワークの学習および評価のため，新たに折り紙点群データセットを作成した．構築したデータセットを用いてネットワークの学習を行い，提案手法で視点位置が推定精度にどのように影響するかを調査した．実験結果から，視点位置によって推定精度が異なり，比較的正確に推定できる位置とできない位置があることがわかった．また，本稿で提案した折り紙を対象とする深層学習ネットワークは，限定的な条件下ではあるが有効に機能していることが確認できた．

今後の課題として，本研究では折る過程が単純な紙飛行機を対象データとして検討を進めたが，より実用的な課題に向けて紙飛行機以外のデータセットを作成し，提案した深層学習ネットワークの評価，および改良を行う必要がある．また，展開図の推測の再検討を通じて，実世界の点群センサを用いることができるように検討を進めていくことも必要である．

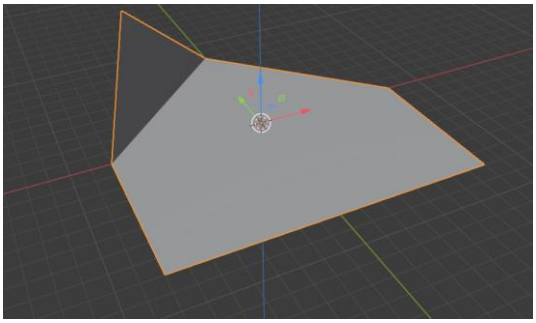

図 １２：目標データ例 (Test Loss が最大の場合)

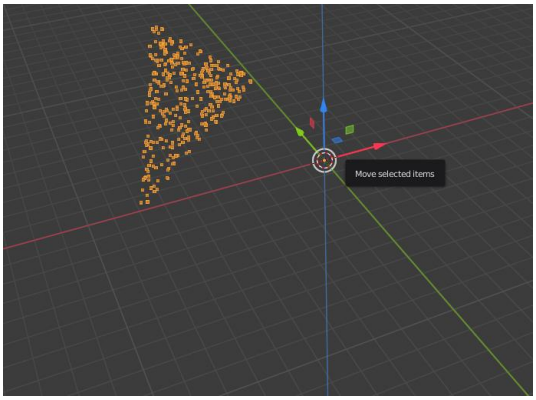

図 １３：入力データ例 (Test Loss が最大の場合)

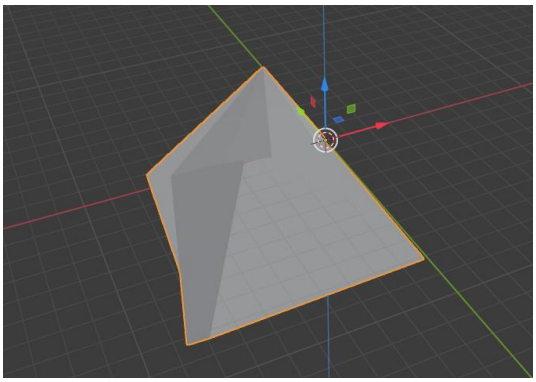

図 １４：ネットワーク出力結果例 (Test Loss が最大の場合)

参考文献

[1] 三谷純. 2 次元バーコードを用いた紙の折りたたみ構造の認識とそのモデル化. 情 報処理学会論文誌, 48(8):2859-2867, 2007.

[2] Akio Namiki and Shuichi Yokosawa. Robotic origami folding with dynamic motion primitives. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5623–5628. IEEE, 2015.

[3] Christof Elbrechter, Robert Haschke, and Helge Ritter. Folding paper with anthropomorphic robot hands using real-time physics-based modeling. In 2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012), pp. 210–215. IEEE, 2012.

[4] Tomohiro Tachi. Simulation of rigid origami. Origami, 4:175–187, 2009.

[5] Ke Liu and Glaucio H Paulino. Merlin: A matlab implementation to capture highly nonlinear behavior of non-rigid origami. In Proceedings of IASS Annual Symposia, volume 2016, pp. 1–10. International Association for Shell and Spatial Structures (IASS), 2016.

[6] Amanda Ghassaei, Erik D Demaine, and Neil Gershenfeld. Fast, interactive origami simulation using GPU computation. Origami, 7:1151–1166, 2018.

[7] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-CNN: Octree-based convolutional neural networks for 3D shape analysis. ACM Transactions on Graphics (TOG), 36(4):1–11, 2017.

[8] Peng-Shuai Wang, Chun-Yu Sun, Yang Liu, and Xin Tong. Adaptive o-CNN: A patch-based deep representation of 3D shapes. ACM Transactions on Graphics (TOG), 37(6):1–11, 2018.

[9] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 652–660, 2017.

[10] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mˆach´e approach to learning 3D surface generation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 216–224, 2018.

[11] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 206–215, 2018.

[12] Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. Learning elementary structures for 3D shape generation and matching. In Advances in Neural Information Processing Systems, pp. 7433–7443, 2019.

[13] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3D-coded: 3D correspondences by deep deformation. In Proceedings of the European Conference on Computer Vision (ECCV), pp. 230–246, 2018.

[14] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4460–4470, 2019.

[15] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In Proceedings of the IEEE International Conference on Computer Vision, pp. 7154–7164, 2019.

[16] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3D object reconstruction from a single image. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 605–613, 2017.

[17] Paolo Cignoni, Claudio Rocchini, and Roberto Scopigno. Metro: measuring error on simplified surfaces. In Computer graphics forum, volume 17, pp. 167–174. Wiley Online Library, 1998.

[18] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An information-rich 3D model repository. CoRR, abs/1512.03012, 2015.

[19] 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 28th IEEE Conference on Computer Vision and Pattern Recognition (CVPR2015), 2015.

[20] Jun Mitani. Development of origami pattern editor (ORIPA) and a method for estimating a folded configuration of origami from the crease pattern. IPSJ Journal, 48(9):3309–3317, 2007.