Vox2C-space:
動作計画のための機械学習に基づくC-spaceの生成

○木南貴志（中部大学）　山内悠嗣（中部大学）

1. はじめに

近年，労働力の不足や人件費の削減などの理由により，ロボットを活用した作業の自動化の需要が高まっている．ロボットによる作業の自動化を実現するには，ロボットアームの動作時の軌道を作成する必要があり，教示と動作計画の 2 種類が存在する．教示は作業者がロボットを操作して動作を記憶し，記憶した動作を連続的に再生するティーチングプレイバックにより作業を再現する．教示には多くの時間が必要になることや，作業内容や環境が変わったときに，教示をやり直す必要があるため，近年では動作計画に関する研究が盛んに取り組まれている．

動作計画は，ロボットが障害物を避けながら初期姿勢から目標姿勢へ達する軌道を計画する問題である．動作計画では Configuration-space(C-space) と呼ばれるロボットの姿勢を表現した空間が用いられる．実空間におけるロボットと障害物が衝突するような姿勢は，C-space 上においてコンフィギュレーション障害物として写像されるため，C-space 上でコンフィギュレーション障害物を回避する経路を生成することで，実空間においても障害物を回避するような動作を生成することができる．全ての姿勢における衝突情報を写像した完全な C-space を作成するには膨大な計算を必要とするため，動作計画の際には完全な C-space を作成することなく，逐次的に衝突判定を行いながら経路を探索する．そのため，サンプリングベース手法などの逐次的に経路を探索する以外の手法を利用することができない問題がある．

そこで，本稿では機械学習に基づいて C-space を生成する Vox2C-space を提案する．Vox2C-space は，エンコーダ・デコーダ型のネットワークで構成された C-space 生成ネットワークである．提案手法は，2 値のボクセルで表現された占有グリッドマップから機械学習により C-space を生成する．直接的に衝突判定の計算を行わないため，C-space を高速に計算することが可能となる．

2. 関連研究

2.1 動作計画法

現在に至るまでに動作計画法に関して盛んに研究されてきた．動作計画法は完全な C-space を必要とする手法と，そうではない手法の 2 種類が存在する．

完全な C-space を必要とする動作計画法として，ポテンシャルベースの手法 [1] が存在する．ポテンシャルベースの手法では，目標姿勢に引力ポテンシャル，障害物に斥力ポテンシャルを仮想的に設置したポテンシャル場を作成することで目標姿勢までの軌跡を生成する．この手法は，障害物の回避と目標姿勢への到達を同時に実行できる一方，ポテンシャル関数の勾配が 0 になる停留現象が生じる可能性がある．また，3 次元以上の多次元空間においては，C-space の作成に膨大な計算量を必要とする．多次元の完全な C-space を作成するには多くの衝突判定が必要になるため高速化した手法が提案されている．Fastron[2] は C-space における衝突領域と非衝突領域の境界線を検出することで，衝突判定の計算回数を削減している．障害物の数が多い環境でも処理時間が増加しない特徴を持つ．

完全な C-space を必要としない動作計画法としてはランダムサンプリングベースの手法が存在する．ランダムサンプリングベースの手法は，C-space 上にランダムにノードをサンプリングし，ノード間を結ぶことで目標姿勢までの軌跡を生成する． 代表的な手法としては Probabilistic Road Map(PRM)[3] や Rapidly-exploring Random Tree(RRT)[4] が存在する．PRM と RRT は，どちらもノードをランダムにサンプリングする手法であるが，前者は事前にサンプリングを行うのに対して，後者は逐次的にサンプリングする．PRM では，事前にランダムにサンプリングされたノード間を結ぶことで軌跡を生成する．RRT では，ランダムにサンプリングされたノードと最も近いノードを結ぶことを繰り返すことで目標姿勢までの軌跡を生成する．ランダムサンプリングベースの手法は，高次元の C-space においても比較的高速に動作することから多軸ロボットの動作計画にも用いられる．一方で，ランダムに経路を生成するため，経路の最適性は保証されない．Chenning らは，このような欠点を改善するために，Graph Neural Network(GNN) を用いた動作計画の手法 [5] を提案した．GNN で，非衝突領域の探索と経路の平滑化を学習することで，従来法と比べて高速に目標姿勢までの最適な経路を探索することが可能となる．

2.2 画像生成

画像生成の分野では，深層学習に基づいて高画質な画像を生成する研究が盛んに取り組まれている．画像生成手法の 1 つである Generative Adversarial Network(GAN)[6] は，偽物の画像を生成する Generator と，入力された画像が本物か偽物かを見分ける Discriminator の 2 つのネットワークによって構成されている．Generator と Discriminator の双方の学習を進めることで高精度な画像の生成が可能となる．GAN を発展させた研究も進められており，その 1 つに pix2pix[7] が提案されている．pix2pix は，2 つのペアの画像から学習した関係性に基づき，1 枚の画像から関係性を考慮した画像を生成する技術である．さらに，2 次元画像から 3 次元データを生成する手法が提案されており，Pix2Vox[8] は，複数視点から撮影した被写体の 2 次元

画像を入力することで，被写体の 3 次元ボクセルを生成する．出力される複数のボクセルを統合することで高精度な 3 次元ボクセルが生成できる．

2.3 提案手法の概要

本研究では，計算に時間を要する衝突判定を回避するために，画像生成技術を応用することで完全な C-space を生成する．本研究の貢献点は以下の通りである．

- 完全な C-space の生成
  C-space を計算するための衝突判定は多大な時間を要する．そこで，本研究では画像生成ネットワークを応用することで衝突判定をすることなく C-space を生成する．
- 異なる空間への写像
  画像生成に関する研究は，ベクトルもしくは画像に基づいて画像を生成する手法が一般的である．本研究は 3 次元データから別次元の全く異質な空間に写像する手法を提案する．

3. 提案手法

提案する Vox2C-space は図 1 のように，占有グリッドマップとして表現される 2 値のボクセルから C-space を生成することを目的としたネットワークである．

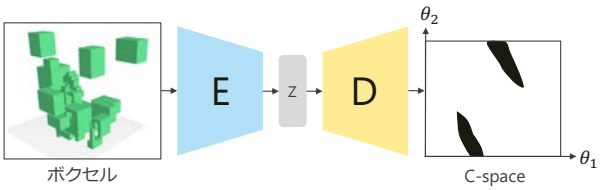

図 1　提案手法のネットワークの概要図.

ネットワークの入力となるボクセルは 2 値で表現され，0 は空間に何も存在しない，1 は何らかの物体が存在することを表す 3 次元の占有グリッドマップとして表現される．C-space はロボットの位置や各関節の変位量等を一般化座標空間に写像した空間である．N 軸のロボットアームの場合，n 番目の関節の変位量を θ_n とすると，N 軸ロボットアームの関節変位は $\boldsymbol{q} = (\theta_1, \theta_2, \cdots, \theta_N)$ となり，C-space 上の 1 点のノードとして表現される．C-space 全体を C_{all}，障害物と衝突，もしくは自己干渉する空間を C_{obstacle} とすると，C-space 上でロボットが動作可能な空間は $C_{\mathrm{free}} = C_{\mathrm{all}} \backslash C_{\mathrm{obstacle}}$ として表現される．また，C-space においてノード $\boldsymbol{q}$ が探索可能な領域は $\boldsymbol{q} \in C_{\mathrm{free}}$ となる．

3.1 ネットワークアーキテクチャ

図 2 に，ロボットの関節の回転に関する分解能が 1 度の C-space を生成するネットワークアーキテクチャを示す．Vox2C-space は，占有グリッドマップとして表現された 2 値のボクセルをエンコーダに入力することで特徴マップを抽出する．そして，得られた特徴マップをデコーダに入力することで C-space を生成する．

エンコーダは，$32^3 \times 1$ のボクセルを入力として，3 次元畳み込み層，インスタンス正規化層，ReLU 層，プーリング層を用いて $4^3 \times 512$ の特徴量を抽出する．エンコーダから出力された特徴量は 3 次元であるため，特徴量を $4^2 \times 2048$ にリサイズする．次に，リサイズした特徴量を入力として，転置畳み込み層，インスタンス

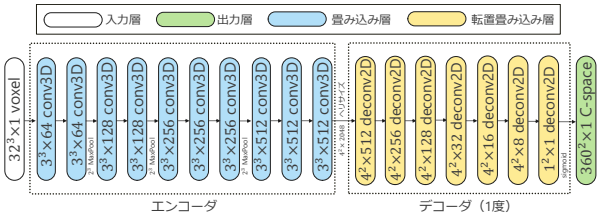

図 2　ネットワークの構成 (C-space の分解能 1 度).

正規化層，ReLU 層と続き，最後にシグモイド層に通して任意のサイズの C-space を生成する．出力される C-space は，C_{free} が 0，C_{obstacle} が 1 として表現され，ロボットアームの各関節の分解能によって C-space のサイズが異なる．

3.2 損失関数

損失関数は，生成された C-space と真の C-space のクロスエントロピー誤差の平均値と正則化項の総和として定義される．C-space のセルの総数を N，衝突領域 C_{obstacle} である確率を p_i，それに対応する真値を gt_i，L1 正則化を L_1，直交正則化を L_{ortho}，パラメータを α，β とした時，損失関数は式 (1) で表現される．

$$L = \frac{1}{N}\sum_{i=1}^{N}[gt_i \log(p_i) + (1 - gt_i)\log(1 - p_i)] + \alpha L_1 + \beta L_{ortho} \quad (1)$$

ネットワークの各ニューロンの重みを $\boldsymbol{w}_i$，重みの総数を N とした時，L1 正則化項と直交正則化項はそれぞれ式 (2)(3) で表現される．

$$L_1 = \sum_{i=1}^{N} |\boldsymbol{w}_i| \quad (2)$$

$$L_{ortho} = \sum_{i=1}^{N} |\boldsymbol{w}_i \boldsymbol{w}_i^{\top} - \boldsymbol{I}| \quad (3)$$

ここで，$\boldsymbol{I}$ は単位行列を表す．

4. 評価実験

提案手法の有効性を確認するために 2 種類の評価実験を行う．1 つ目は，生成される C-space の精度を比較する．2 つ目は，従来の C-space を用いた動作計画と提案手法によって生成された C-space を用いた動作計画の処理時間を比較する．

4.1 データセット

ロボットシミュレータ Gazebo[9] を用いて，データセットの作成を行う．ロボットには 2 軸ロボットアーム RRBot[1] を 1/2 のサイズに縮小したものを用いる．各軸の回転範囲は $[-180, 180)$ となる．

4.1.1 環境

シミュレーション空間にロボットと障害物を生成し，上空に設置した 4 つのカメラで 2.3[m^3] の範囲の 3 次元データを点群として撮影し，ボクセルと C-space を作成する．障害物は，各辺の長さの範囲が [0.1m, 0.35m] の直方体として表現し，範囲内のランダムな位置に 15 個生成する．

[1] https://github.com/ros-simulation/gazebo_ros_demos

4.1.2 ボクセルと C-space

ボクセルは，4 つのカメラから撮影された点群を統合した後に，32^3 のサイズにボクセル化したものを使用する．C-space は，ロボットの各姿勢において OctoMap[10] を用いた衝突判定を実施し，その結果を記録したものを使用する．今回は，ロボットアームの各関節の分解能が 1 度，5 度，10 度の C-space を作成した．また，データセットはボクセルと C-space をペアとして，各分解能ごとに 30,000 データ作成した．データセットの例を図 3 に示す．

(a) Gazebo　(b) ボクセル（入力）　(c) C-space（出力）

図 3　データセットの例．

4.2 実験結果

4.2.1 C-space の精度の評価

生成された C-space の精度を評価する．ロボットアームの各関節の分解能を 1 度，5 度，10 度と変化させた時に生成される C-space の精度を比較する．評価指標には式 (1) によって算出された損失と IoU の 2 種類を使用する．

提案手法における，損失と IoU を表 1 に示す．表 1 の結果より，損失，IoU 共に分解能による差がほとんどないことが分かる．生成された C-space を図 4 に示す．どのような状況においても，C-space におけるコンフィギュレーション障害物を大まかに生成できていることがわかる．ただし，コンフィギュレーション障害物の境界付近の精度が低い．これは，入力が 32^3 の分解能が低いボクセルであるため，抽出できる特徴に限界があることが要因と考えられる．提案手法により生成した C-space を用いた経路探索は，障害物の回避を 100%保証することはできないため，求めた軌跡が障害物に衝突していないことを確認する必要がある．なお，計画した経路は C-space 全体と比べると僅かな割合であるため，衝突判定に必要な追加の計算量は少ない．衝突する姿勢が見つかった場合は，既存の動作計画手法により，経路を部分的に修正することで衝突の回避を保証できる．

表 1　各分解能における生成された C-space の比較．

分解能	1deg	5deg	10deg
損失	1.958	1.902	2.056
IoU	0.793	0.796	0.790

4.2.2 動作計画に要する時間の比較

初期姿勢と目標姿勢が異なる 3 シーケンスにおける動作計画の処理時間を比較する．この際，軌跡を追従する時間は処理時間に含めない．動作計画手法には RRT を使用する．RRT はサンプリングベースの手法であるため，乱数による結果のばらつきを抑制するために同条件で 5 回の実験を行い，その平均処理時間を比較する．実験は，メモリが 16GB，CPU が AMD Ryzen7 5800X，GPU が NVIDIA GeForce RTX 3070 の PC を用いた．

各シーケンスにおける処理時間の結果を表 2 に示す．表 2 より，従来法に比べて平均で 96.2%の処理時間を削減できていることが分かる．経路計画において従来手法の方が大きく処理時間が増加しているが，これは OctoMap による衝突判定が要因と考えられる．特に，今回の実験のように障害物の数が多い環境では衝突判定の時間が増加する．そのため，本手法は障害物が多い環境で性能を発揮しやすいと考えられる．

表 2　動作計画に要する時間．

	平均処理時間 [msec]		
シーケンス	1	2	3
従来手法	1775.8	8414.2	1277.2
提案手法 (1deg)	52.7	361.1	23.3

図 5 に各シーケンスにおける経路の探索結果を示す．正解の C-space と提案手法により生成した C-space は類似しているため，経路探索結果も同じような結果が得られていることがわかる．今回の実験では経路探索法に RRT を採用したが，C-space を利用する他の手法においても適用可能である．

5. おわりに

本研究では機械学習による C-space の生成手法を提案した．機械学習によって C-space を生成することで，従来の動作計画と比較して処理時間を削減することができた．今後は，本手法の更なる高精度化や時系列データへの対応に取り組む予定である．

参 考 文 献

[1] Y. Koren *et al.*, “Potential field methods and their inherent limitations for mobile robot navigation”, *ICRA*, vol. 2, pp. 1398–1404, 1991.

[2] N. Das *et al.*, “Learning-based proxy collision detection for robot motion planning applications”, *IEEE Trans. on Robotics*, vol. 36, no. 4, pp. 1096–1114, 2020.

[3] L. Kavraki *et al.*, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces”, *IEEE Trans. on Rob. and Autom.*, vol. 12, no. 4, pp. 566–580, 1996.

[4] S. M. Lavalle, “Rapidly-exploring random trees: A new tool for path planning”, *technical report*, 1998.

[5] C. Yu *et al.*, “Reducing collision checking for sampling-based motion planning using graph neural networks”, *Neural IPS*, vol. 34, pp. 4274–4289, 2021.

[6] I. J. Goodfellow *et al.*, “Generative adversarial nets”, *Neural IPS*, vol. 27, pp. 2672–2680, 2014.

[7] P. Isola *et al.*, “Image-to-image translation with conditional adversarial networks”, *CVPR*, pp. 5967–5976, 2017.

[8] H. Xie *et al.*, “Pix2vox: Context-aware 3d reconstruction from single and multi-view images”, *ICCV*, pp. 2690–2698, 2019.

[9] N. Koenig *et al.*, “Design and use paradigms for gazebo, an open-source multi-robot simulator”, *IROS*, vol. 3, pp. 2149–2154, 2004.

[10] A. Hornung *et al.*, “Octomap: an efficient probabilistic 3d mapping framework based on octrees”, *Auton. Robot.*, vol. 34, no. 3, pp. 189–206, 2013.

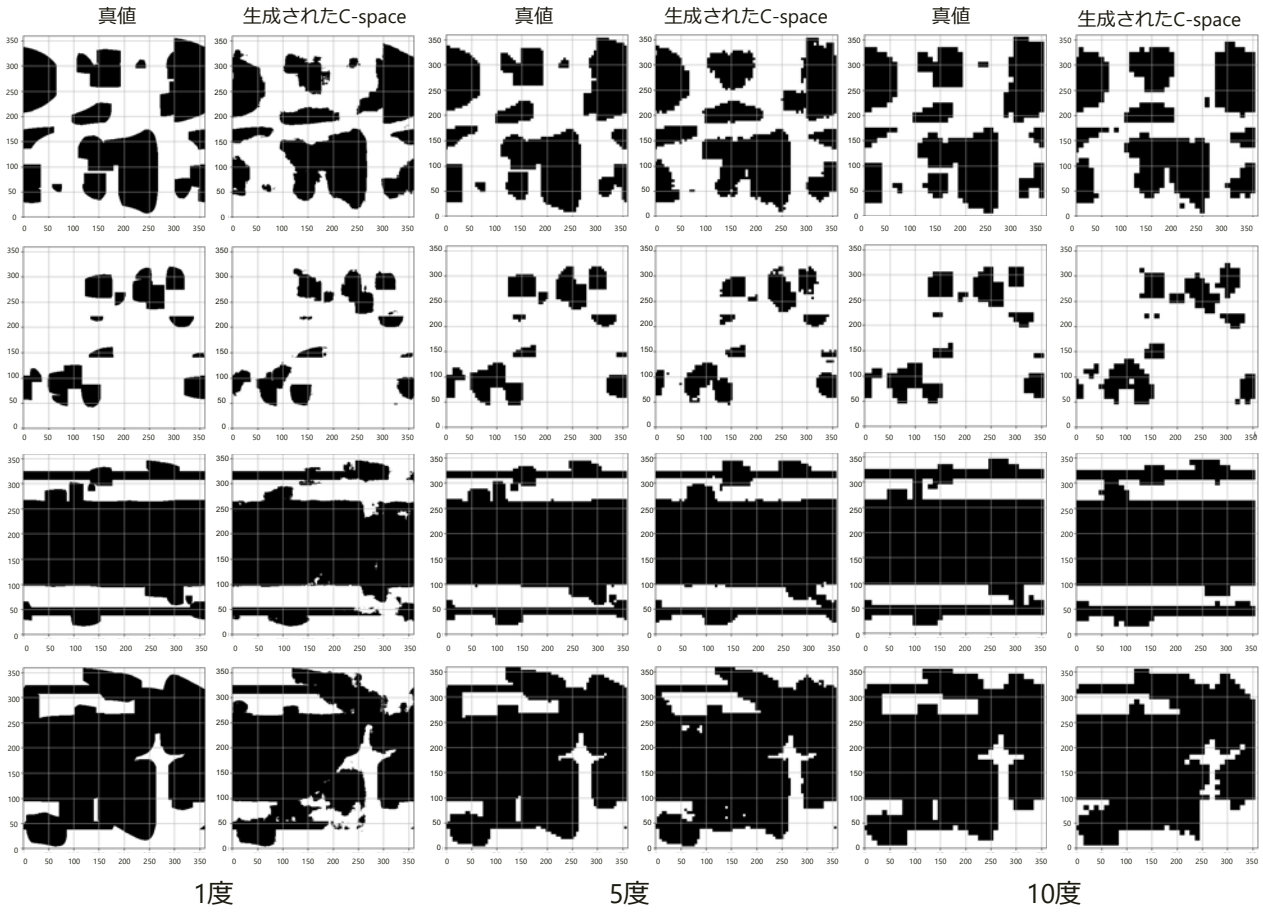

図 4　生成された C-space.

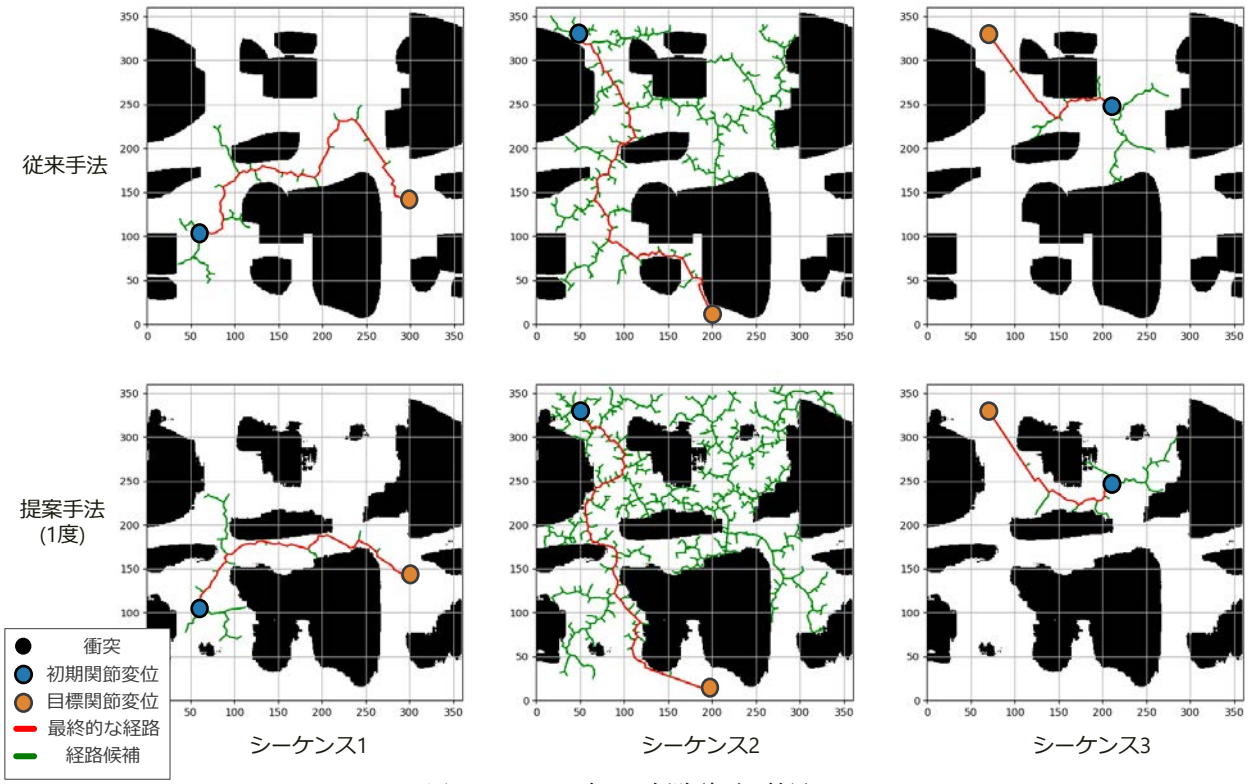

図 5　C-space 上での経路計画の結果.