IIEEJ 論文

キーポイントパッチ抽出法を用いた高能率な進化計算による3次元点群レジストレーション

植西一馬[†](正会員)　サンドバル・ハイメ[†](学生会員)　岩切宗利[††](正会員)
田中清[†††](フェロー)

†信州大学総合工学系研究科，††防衛大学校情報工学科，†††信州大学学術研究院 (工学系)

3D Point Cloud Registration by Efficient Evolutionary Computation with Keypoint Patches Extraction

Kazuma UENISHI[†](*Member*), Jaime SANDOVAL[†](*Student Member*), Munetoshi IWAKIRI[††](*Member*), Kiyoshi TANAKA[†††](*Fellow*)

†Interdisciplinary Graduate School of Science and Technology, Shinshu University,
††Department of Computer Science, National Defense Academy of Japan,
†††Academic Assembly (Institute of Engineering), Shinshu University

〈あらまし〉 3次元点群のレジストレーションは，モデリングやオブジェクト認識などの応用技術の前処理としてよく使われる．進化計算による方式は，初期位置に依存せず高精度にレジストレーションできるという特長をもつが，他の方式と比較して計算量が大きいという問題がある．この原因は，評価関数を求めるための最近傍探索回数が多いためと考えられる．探索回数の削減には点群のリサンプリングがよく用いられるが，これはレジストレーション精度が低下することが多い．そこで，キーポイントの周囲の点のみを抽出した点群 (キーポイントパッチ) を進化計算によるレジストレーションに用いる手法を提案する．また，更なる高精度化及び高速化を実現するため，境界の活用によるオクルージョンの悪影響の低減，キーポイントパッチの個数削減，キーポイントパッチのリサンプリングによる点数の削減を提案する．8種類のデータセットを用いた実験結果から，提案方式は従来方式と比較して，レジストレーション精度を維持しつつ約100倍以上高速化できることを確認した．

キーワード：3次元レジストレーション, 進化計算, 特徴点検出, キーポイントパッチ

<**Summary**> 3D point cloud registration is often used as preprocessing of several applications such as modeling and object recognition. Evolutionary computation based registration has the merit that it can register with high accuracy without depending on the initial position, however there is a problem that the computational cost is larger than the other methods. This reason is considered to be that the number of nearest neighbor searches for finding the evaluation function is computationally expensive. To reduce the number of searches, resampling of point clouds is commonly used, but registration accuracy often decreases. Therefore, we propose an evolutionary computation based registration method which extracts only points around the keypoint. This keypoint patches achieve that maintains registration accuracy and reduces computational costs. Moreover, in order to realize further higher accuracy and speed, we propose to reduce the adverse effects of occlusion by avoiding boundaries, reducing the number of keypoint patches, resampling points in the keypoint patches. Experimental results using 8 kinds of datasets confirmed that the proposed method can speed up about over 100 times while maintaining the registration accuracy as compared with the conventional methods.

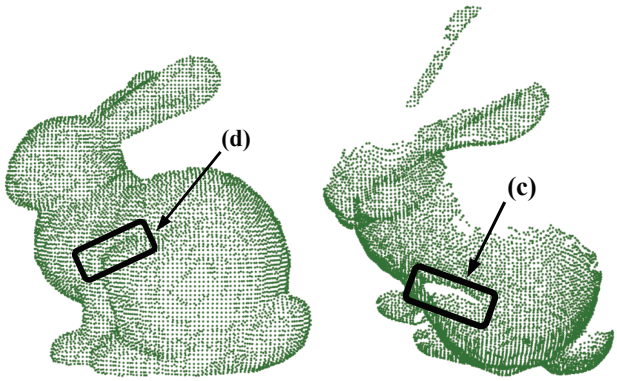
Keywords: 3D registration, evolutionary computation, keypoint detector, keypoint patches

1. はじめに

3次元点群の処理技術は，モデリング[1)]やオブジェクト認識[2)]などに応用されており，コンピュータビジョンの重要な技術のひとつであるといえる[3)]．また，安価な3次元センサの普及により，点群の処理技術の需要は増してきている．しかしながら，点群の処理には解決すべき複数の問題がある．

(a) Front view (b) Top view

図 1 “bun000” の点群
Fig. 1 The point cloud of “bun000”



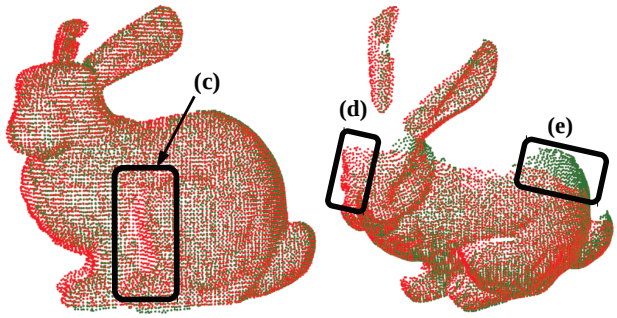
(a) Front view (b) Top view

図 2 “bun045” の点群
Fig. 2 The point cloud of “bun045”

まず，点群は離散的に莫大な数の点を配置することによって形状を表現しているため，ノイズが全くなくとも計算量が多くなることが多い．更に，センサやオブジェクトの移動により，点群を成す点分布は大きく変動する．例えば，オブジェクトの裏側や，別のオブジェクトによって影となった領域は，点分布が欠損する．この欠損した領域をオクルージョン領域と呼ぶ[4]．オクルージョンのほかにも，センサの解像度によっては点分布の密度が乏しく，またノイズによって変動するため，形状を十分に表現できていない場合もある．

図 1 と**図 2** は，The Stanford 3D Scanning Repository[5] にある Stanford Bunny[6] の “bun000” 及び “bun045” の点群を表示したものである．“bun000” におけるオブジェクトとセンサの位置関係を基準に，右 45 度の位置からセンシングしたものが “bun045” である．それぞれ (a) の図はセンサの視点から，(b) の図は上方に視点をおいて点群の様子を表示した．両点群ともに，センサに露出しないオブジェクトの裏側は点分布が欠損している．また，図 2(c) に示す部分は，後ろ足の膝の部分 (図 2(d)) が影となり，一部が欠損している．

先に述べた問題のうち，オクルージョンや低密度の問題を低減する方式として，複数の点群を重ねあわせて 1 つに統合するレジストレーションがある．これにより点群の欠損を補完し，さらに高解像度化できるため，モデリングなどの応用技術の前処理としてよく用いられる．重ね合わせの基準となる座標系の点群をターゲット点群 $\mathbb{T} = \{\boldsymbol{p}_i^t | i = 1, ..., N_t\}$，座標系変換される点群をソース点群 $\mathbb{S} = \{\boldsymbol{p}_i^s | i = 1, ..., N_s\}$ と呼ぶ．$\boldsymbol{p}_i^t$ はターゲット点群を，$\boldsymbol{p}_i^s$ はソース点群を構成する点の位置座標を示す．N_t はターゲット点群の N_s はソース点群の点の個数である．



(a) Front view (b) Top view

図 3 レジストレーションした点群
Fig. 3 Registered point cloud

図 3 は，“bun000” をターゲット点群とし，“bun045” を正確にレジストレーションした結果である．図 3(c),(d),(e) に示す部分のようにオクルージョン領域は埋め合わされ，さらにオクルージョン以外の部分は赤点と緑点が混ざり合って解像度が向上した．このようなレジストレーションの実現には，ターゲット点群の座標系への変換パラメータを高精度に得る必要がある．3 次元センサの移動量や回転量を測定する外部センサなしでレジストレーションするためには，点群間の位置関係や点分布のみから変換パラメータを推定しなければならない．本稿では，色情報の取得機能をもたないセンサで取得した，単に点の位置情報の集まりである点群のレジストレーションについて検討する．

点群のみから変換パラメータを推定する方式には，Iterative Closest Point (ICP)[7]，特徴点による方式[8]，進化計算による方式[9] が挙げられる．ICP は最近傍探索により得た解から変換パラメータを推定し，そのレジストレーション精度を評価して座標系変換を繰り返す再帰的方式である．最もよく用いられる方式であるが，レジストレーションの精度や計算量は点群の初期配置に依存するという欠点が存在する．この改善のため，局所形状や主曲率などを ICP に用いる手法[10] が提案されているが，局所解の付近に解を得た場合にこれを回避できない．特徴点による方式は，抽出した局所特徴による解から変換パラメータを推定し，一度だけ座標系変換する方式である．これは最も計算量が少ないが，局所特徴は点分布の変化に敏感であり，検出位置の変動や誤対応の影響を強く受ける．これらの方式は相互に欠点を補うため，特徴点による方式を前処理とし，その後 ICP をする Coarse-to-Fine アプローチ方式も存在する[11]．しかしながら，特徴点による

方式が局所解の付近に解を得た場合，その局所解を回避できない．すなわち，ICP をベースとした方式は，局所解に近づいた際にこれを回避できないという問題が残る．

進化計算による方式 (Evolutionary Computation Registration: ECR) は，進化計算により得た変換パラメータを用いる再帰的方式である．解を得る手法が進化計算アルゴリズムによるため，付近の局所解を回避しつつ，全体最適解を探索することができる．すなわち，ECR は ICP などとは異なり，初期位置に依存せず高精度な変換パラメータを推定できる方式であるといえる．しかしながら，最近傍探索の回数が ICP よりもはるかに多いため，莫大な計算量を必要とする問題がある[10]．

ECR の問題を解決するため，Uenishi らは ECR の計算量を削減するキーポイントパッチ (KeyPoint Patches: KPP) 法を提案した[12]．これは，ソース点群から検出したキーポイントの周囲の点をパッチ状に抽出し，ECR に用いる方式である．KPP を成す点数は元の点群よりも少ないので，最近傍探索の回数を削減することができる．実験により，KPP 法は ECR の精度を保ったまま計算量を大幅に削減できることを確認できたが，オクルージョンの影響により最適解の付近で局所解に収束してしまう場合もあった．この対策として，点群の境界点を活用することでオクルージョンの影響を低減できることを確認した[13]．また，KPP の個数や，KPP を成す点数の削減効果についても検討し，精度を維持しつつ計算量削減の効果を確認した[13]．本稿では，KPP 法に関する提案[12,13] を全て総括し，その詳細について述べる．さらに，実験用データセットの追加及びノイズを付与した実験の追加により，提案方式の性能を包括的に評価した．

本稿の構成は次のとおりである．2 章では，ECR の詳細と，本稿で用いる進化計算アルゴリズムについて述べる．3 章において，提案方式である KPP 法の詳細と，境界の活用，KPP 個数の削減及び点数の削減などの提案方式について述べる．4 章では，提案方式の性能評価のための実験について，その結果と考察を述べる．最後に，本稿のまとめと課題を述べる．

2. 関 連 研 究

2.1 進化計算レジストレーション

一般的に，進化計算は次の処理を適用する[9]．

Step 1 ランダムな値の要素をもつ個体 $\boldsymbol{k}_i^h$ を，N_p 個生成する．i は個体数の，h は世代数のインデックスである．

Step 2 評価関数により，現世代 g における $\boldsymbol{k}_i^g$ の適応度 FS_i^g を求める．

Step 3 次項で述べるような進化計算アルゴリズムによって $\boldsymbol{k}_i^g$ の値を変化させ，進化個体 $\boldsymbol{t}_i^{g+1}$ を得る．

Step 4 評価関数により，$\boldsymbol{t}_i^{g+1}$ の適応度 FS_i^{g+1} を求める．

Step 5 FS_i^{g+1} が FS_i^g よりも適している場合，$\boldsymbol{k}_i^g$ と FS_i^g の値を更新する．

Step 6 終了条件に達するまで，Step 3 から 5 を繰り返す．

Step 7 最も適した FS_i^g を示す $\boldsymbol{k}_i^g$ を解として取り出す．

終了条件には，世代数や適応度に閾値を設定することがよく用いられるが，実際の処理時間で打ち切る場合もある．進化計算をレジストレーションに適用するため，個体や評価関数を以下のように設計する．

求めるべき変換行列 $\boldsymbol{T_e}$ は，4×4 の行列であるが，これは原点からの 3 軸回転と，3 軸並進から求めることができる．すなわち，ある個体 $\boldsymbol{k}$ は，これらを包含した 6 次元のベクトルで表現できる．ある個体 $\boldsymbol{k}$ からなる $\boldsymbol{T_k}$ で $\mathbb{S}$ を写像したとき，その FS は式 (1) により求める[14]．

$$FS(\boldsymbol{T_k}\mathbb{S}, \mathbb{T}) = \mathrm{Median}(d_i)\ ,\ (i = 1, ..., N_s) \tag{1}$$

Median$(\cdot)$ は中央値を求める関数，d_i は最小自乗距離であり，式 (2) で定義される．

$$d_i = ||\boldsymbol{T_k}\boldsymbol{p}_i^s - \boldsymbol{p}_{cl}^t||^2 \tag{2}$$

ここで cl は，$\boldsymbol{T_k}$ で座標系変換した $\boldsymbol{p}_i^s$ から最も近傍にある $\mathbb{T}$ の点のインデックスである．評価関数に中央値を用いる理由は，オクルージョンの影響を低減するためである．

文献 9) によると，ECR はランダムに個体を生成するため，局所解に陥りにくくレジストレーションが成功しやすい．しかしながら，レジストレーションの精度や処理時間は，実験により比較評価されていない．文献 10) には，遺伝的アルゴリズムを採用した ECR[15] で比較実験し，ICP など ECR 以外の方式と比較して莫大な処理時間を要することが示されている．この原因には，最近傍探索の回数が挙げられている．1 つの個体の FS を求めるのに，N_t 点に対し N_s 回の探索が必要であり，さらに個体数と世代数が計算量に乗算される．一方で，ICP では N_t 点に対する N_s 回の探索が，50 回程度で収束に至るとされている[7]．

2.2 Differential Evolution

進化計算アルゴリズムは複数存在するが，そのうちの 1 つである，Differential Evolution (DE)[16] について述べる．DE を用いた ECR は，文献 9) の最も条件の厳しいデータセットにおいて，レジストレーション精度の平均値が他のアルゴリズムよりも高かった．そこで本稿では，この DE に着目した．

DE において，現世代 g の個体 (ターゲットベクトル)$\boldsymbol{k}_i^g$ の進化は，他の 3 つの個体 $\boldsymbol{k}_{r_1}^g, \boldsymbol{k}_{r_2}^g, \boldsymbol{k}_{r_3}^g$，突然変異係数 $F \in [0, 2]$ 及び交叉係数 $C \in [0, 1]$ によって算出される．$r_1, r_2, r_3 \in [1, N_p]$ は，母集団からランダムに選択された個体のインデックス $(r_1 \neq r_2 \neq r_3 \neq i)$ である．このとき，進化個体を生成するための突然変異ベクトル $\boldsymbol{m}_i^{g+1}$ を式 (3) によって得る．

$$\boldsymbol{m}_i^{g+1} = \boldsymbol{k}_{r_1}^g + F(\boldsymbol{k}_{r_2}^g - \boldsymbol{k}_{r_3}^g) \tag{3}$$

次に，$\boldsymbol{k}_i^g$ と $\boldsymbol{m}_i^{g+1}$ を交叉して，進化個体 (トライアルベクトル)$\boldsymbol{t}_i^{g+1} = \{t_{(j,i)}^{g+1} | j = 1, ..., N_j\}$ を式 (4) によって得る．

$$t_{(j,i)}^{g+1} = \begin{cases} m_{(j,i)}^{g+1} & \text{if } r_4(j) \leq C \text{ or } r_5(i) = j \\ k_{(j,i)}^{g} & \text{otherwise} \end{cases} \quad (4)$$

j はベクトルの要素のインデックス，$r_4(j) \in [0,1]$ は j ごとに得る乱数，$r_5(i) \in [0, N_j]$ は i ごとに得る整数乱数である．N_j は個体の要素の数であり，本稿の ECR においては $N_j = 6$ である．

図 4 は，トライアルベクトル生成の例を示す．この図では，トライアルベクトルの $j = 2, 4, 5$ がターゲットベクトル，$j = 1, 3, 6$ が突然変異ベクトルによって構成されている．

3. 提案方式

3.1 KeyPoint Patches (KPP) 法

ECR を高速化するためには点群を構成する点数，個体数，世代数を削減する必要があるといえる．しかしながら，個体数の削減は局所解に陥る可能性を高め，世代数の削減は最適解への収束途中で処理を終了する可能性がある．そこで，個体数と世代数を確保したまま高速化するため，我々は点数の削減に着目した．リサンプリングによる N_t または N_s の削減は，レジストレーション高速化の前処理としてよく用いられる[11]．しかし，FS の算出に用いる点数も同様に削減されるため，レジストレーション精度とのトレードオフとなる．ECR は，FS の値によって個体の進化を決定するため，この精度を維持しなければ最適解に収束できないと考えられる．

我々は FS の算出に極力影響を与えず，点数を削減するためにキーポイントを活用した[17]．キーポイントとは，複数の点群間で極力同じ位置にありつづけ，他の点と弁別しやすい場所にある点である[8]．前者の性質を再現性，後者を弁別性という．これらを保持した点を検出するため，周囲の点の形状から突出値を算出するアルゴリズムが一般的に用いられている[18]．そこで，$\mathbb{S}$ からキーポイントを検出し，その周囲にある点のみを抽出する KeyPoint Patches (KPP) 法を提案する[17]．抽出する支持半径は，r_{kpp} により指定する．特徴的形状を有する点のみを活用することで，FS を効率よく算出できる．KPP の抽出は $\mathbb{S}$ に限定し，$\mathbb{T}$ からは抽出しない．これは，キーポイントの再現性が低い場合，対応箇所に KPP が抽出されるとは限らないためである．

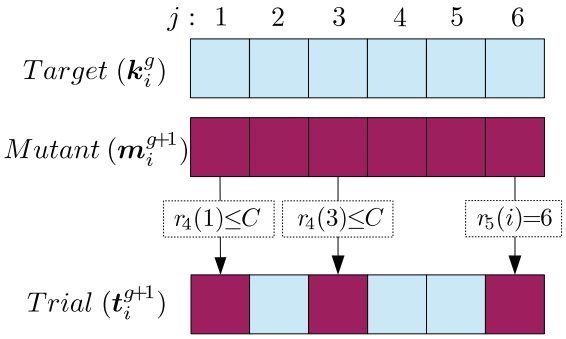
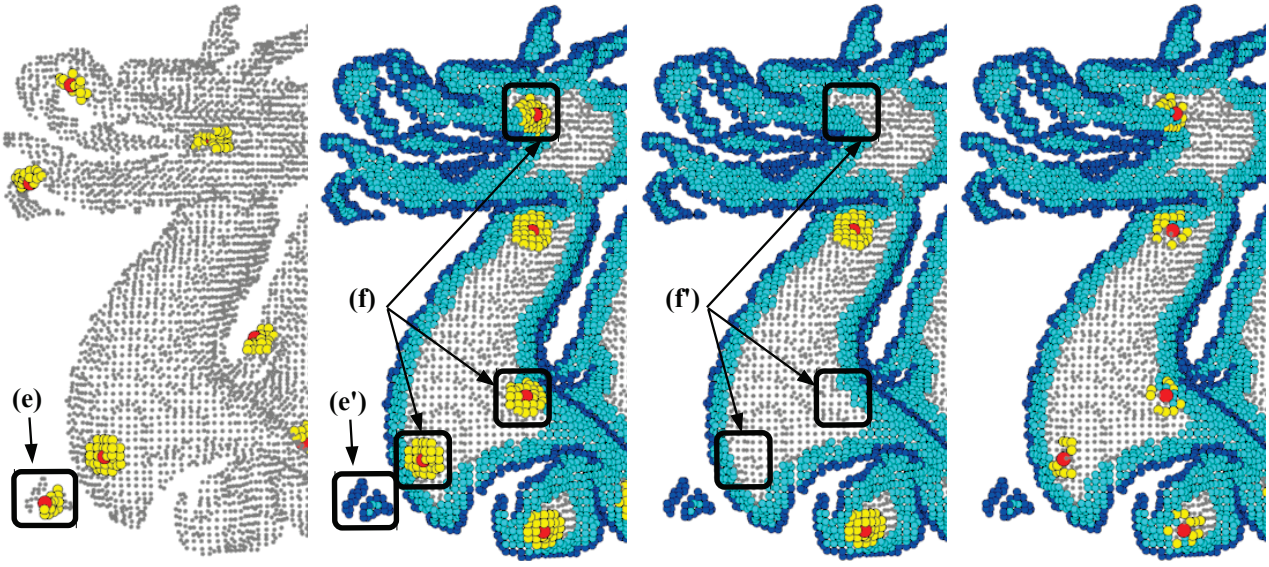


図 4 トライアルベクトルの生成例
Fig. 4 Example of trial vector creation

図 5 (a) の例では，点群から検出したキーポイントを赤点，抽出した KPP を黄点で示している．キーポイント検出には Intrinsic Shape Signatures (ISS)[19] を使用した．この図では，点群全体から 10 個のキーポイントを検出し，点群全体の左半分のみを表示している．もとの点群が 7512 点であるのに対し，KPP は 198 点 (全体の約 2.6%) である．

3.2 KPP 法の主要素

KPP 法は，キーポイント検出とその周囲の点の抽出から成る．前者は KPP の位置と個数を定め，後者は KPP を成す点数を定める．これら 3 つの主要素が，KPP 法の性能を



(a) Original KPP (b) KPP-BR (c) Reduction of KPP (d) Resampling in KPP

図 5 KPP 抽出の例
Fig. 5 Examples of KPP extraction

左右する．そこで，各要素ごと KPP 法に与える影響を考察し，それぞれの改良手法を示す．

3.2.1 KPP の位置

KPP 法によって FS 算出時の最近傍探索回数を削減できるが，d_i の数も減少するため，オクルージョンの影響を受けやすくなる．特に，ある KPP が $\mathbb{T}$ のオクルージョン領域に抽出された場合には，この KPP における d_i の値が大きくなるため，局所解に収束してしまう[12]．例えば図 5 (e) のような，オクルージョンに囲まれた部分に抽出された KPP は，$\mathbb{T}$ 側もオクルージョンである (重なる点がない) 場合が多いといえる．そこで，我々はオクルージョンの性質に着目した．図 3 に示したように，オクルージョンはセンサ側への露出の有無によって生じる．そのため，表面形状が閉じていない領域，すなわち境界付近にはオクルージョンが存在する．この性質を利用し，境界付近の点をキーポイント検出に用いないことで，オクルージョンの影響を回避する方式を提案した[13]．本手法を適用した KPP 法を，KPP-BR (Boundary Removal) 法と呼称する．

図 5(b) は，KPP-BR の抽出位置を示す．境界点は Rusu の手法[20] で抽出し，青色の点で表示した．これは，支持半径 r_{ne} から得た法線ベクトルを活用し，ある点の支持半径 r_{be} 内にある点分布，角度閾値 Th_{be} を用いて全点を境界か否か判定する手法である．オクルージョンから KPP を離隔するため，境界点から半径 r_{iso} 内にある点も境界とみなす．図 5(b) の水色の点が，r_{iso} 内にあるオフセットのための境界点である．その後，境界を除去した点群からキーポイントを検出し，KPP は形状の完全性を確保するため元の点群上からパッチ上に点を抽出する．図 5(a) と図 5(b) を比較すると，図 5(e') のようなオクルージョンに接する KPP がないことが確認できる．このとき，KPP-BR 法による KPP を成す点数は 284 点 (全体の約 3.8%) であった．もとの KPP 法による KPP より点数が増加した理由は，KPP がオクルージョンによって欠けることなく r_{kpp} 内の点を抽出したためである．

3.2.2 KPP の個数

ISS キーポイントは，点の周囲の形状から算出される突出値をもとに検出される．しかし，単に突出値が大きい順に取り出した場合，点群のある箇所にキーポイントが集中する可能性がある．これを避けるため，指定した支持半径 r_{nms} 内で最大の突出値をもつ点のみをキーポイントとして検出する，Non Maxima Suppression (NMS) がよく用いられる[18]．同様に KPP も，少ない数で全体に分散していることが望ましいことから，r_{nms} を用いて KPP の個数を削減する手法を提案する[13]．

図 5(c) は，図 5(b) の r_{nms} を広げて KPP を 4 個に制限した例である．図 5(f) の位置に抽出されていた KPP が，図 5(f') の領域に抽出されていない．これは，図 5(c) で残った KPP を成すキーポイントの突出値が，消えた KPP のものよりも高かったためである．図 5(c) における全 KPP を成す点数は 115 点であり，これは図 5(b) の全 KPP の点数の約 40%，元の点群の約 1.5%であった．

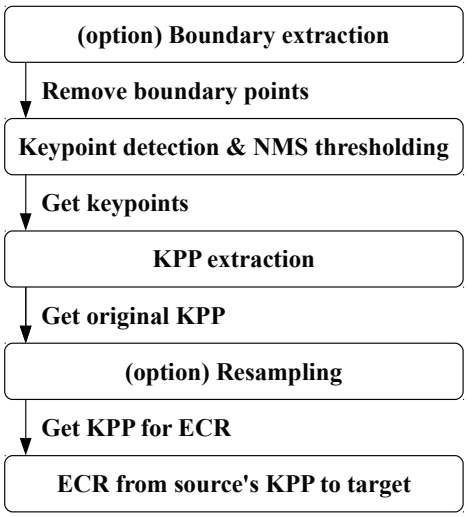


図 6 提案の ECR 処理フロー
Fig. 6 Proposed ECR process flow

3.2.3 KPP を成す点数

KPP を成す点数は，KPP を抽出する範囲や，抽出する密度によって変化する．本研究では抽出する範囲を一定とし，密度をリサンプリングによって変化させることによって，レジストレーションの高速化を KPP 法にも適用する[13]．

図 5(d) は，図 5(b) の KPP を 50%の割合でランダムに間引いた例である．図 5(d) における全 KPP を成す点数は 109 点であり，これは図 5(b) の全 KPP の約 38%，元の点群の約 1.4%であった．

3.3 提案方式の処理手順

図 6 は，提案方式の ECR 処理フローを示す．まず KPP-BR 法を適用する場合は，境界を抽出して点群から削除する．次にキーポイント検出と NMS 処理により KPP の位置を決定し，支持半径内の点を抽出する．点数を更に削減する場合は，KPP をリサンプリングする．その後の ECR は基本的に従来と同様である．ただし，KPP の点群を $\mathbb{K} = \{\boldsymbol{p}_i^k | i = 1, ..., N_k\}$ とするとき，提案方式における FS の算出には式 (1) を用いず，式 (5) を用いる．

$$FS(\boldsymbol{T_k}\mathbb{K}, \mathbb{T}) = \mathrm{Mean}(d_i)\ (i = 1, ..., N_k) \tag{5}$$

ここで，Mean(·) は平均値を返す関数である．中央値を使用しない理由は，一部の KPP だけが $\mathbb{T}$ に重なり，そのまま局所解に陥ることを避けるためである．

本研究では，進化計算のアルゴリズムに Self-Adaptive Differential Evolution (SADE)[21] を採用した．これは DE の派生系であり，係数 F と C を固定せずに，式 (6) と式 (7) を用いて個体ごとに更新していくアルゴリズムである．

$$F_i^{g+1} = \begin{cases} F_l + r_6 \times F_u & \text{if } r_7 < \tau_1 \\ F_i^g & \text{otherwise} \end{cases} \tag{6}$$

表 1 データセットの詳細
Table 1 Dataset detail

ID	Attribute	File name	Voxel [mm]	mr [mm]	Points	Angle [deg]	Overlap [%]	View
$\mathbb{A}_1$	Target	ArmadilloStand_180	2.0	1.3	5292	-	-	Fig.7(b)
$\mathbb{A}_2$	Source	ArmadilloStand_210	2.0	1.3	5095	+30	76.5	Fig.7(a)
$\mathbb{A}_3$	Source	ArmadilloStand_150	2.0	1.4	5559	−30	68.5	Fig.7(c)
$\mathbb{B}_1$	Target	bun000	2.0	1.4	7133	-	-	Fig.7(e)
$\mathbb{B}_2$	Source	bun045	2.0	1.4	6813	+45	86.1	Fig.7(d)
$\mathbb{B}_3$	Source	bun315	2.0	1.3	6831	−45	74.5	Fig.7(f)
$\mathbb{D}_1$	Target	dragonStandRight_0	2.0	1.3	7155	-	-	Fig.7(h)
$\mathbb{D}_2$	Source	dragonStandRight_24	2.0	1.3	6312	+24	89.1	Fig.7(g)
$\mathbb{D}_3$	Source	dragonStandRight_336	2.0	1.3	7512	−24	81.1	Fig.7(i)
$\mathbb{H}_1$	Target	happyStandRight_0	2.0	1.3	5560	-	-	Fig.7(k)
$\mathbb{H}_2$	Source	happyStandRight_24	2.0	1.3	5304	+24	79.6	Fig.7(j)
$\mathbb{H}_3$	Source	happyStandRight_336	2.0	1.3	5600	−24	79.7	Fig.7(l)

$$C_i^{g+1} = \begin{cases} r_8 & \text{if } r_9 < \tau_2 \\ C_i^g & \text{otherwise} \end{cases} \tag{7}$$

$r_6, r_7, r_8, r_9 \in [0,1]$ は乱数，F_l, F_u はコントロールパラメータ，τ_1, τ_2 は更新率である．これらのパラメータは，$F_l = 0.1$，$F_u = 0.9$，$\tau_1 = \tau_2 = 0.1$ が適当であるとされている[21]．SADE は，F と C を適応的に定めることで，コントロールパラメータを少なくしつつ，DE と同等以上の最適解を導出できる利点がある[21]．

4. 実験と考察

4.1 実験データセット

本研究における提案方式の性能評価のため，The Stanford 3D Scanning Repository の Armadillo($\mathbb{A}$)，Stanford Bunny($\mathbb{B}$)，Dragon($\mathbb{D}$)，Happy Buddha($\mathbb{H}$) を使用して実験した．正面からセンシングされた点群を $\mathbb{T}$ とし，その左右の異なる位置からセンシングされた 2 つの点群を $\mathbb{S}$ とする．これらは，実験上の処理時間短縮のため，事前に全ての点群を 1 辺 2.0mm のボクセルでリサンプリングした．ボクセルリサンプリングは Point Cloud Library (PCL)[22] に実装されたものを使用した．これは，ボクセル内にある点の重心点を取り出すもので，ローパスフィルタの効果がある．

データセットの詳細を**表 1** に，点群の様子を**図 7** に示す．表中の単位 mr は Mesh Resolution であり，最近傍点間の距離の中央値により求めた．mr が小さいほど点の密度が高くなり，形状の表現が精緻となる．単位 Points は，点群を構成する全点数である．単位 Angle は，センシング時におけるオブジェクトの Y 軸回転度数 ($\mathbb{T}$ との相対角度) である．単位 Overlap は，ソース点群を真値で変換したときのターゲット点群に対する重なる点数の割合であり，値が大きいほどオクルージョン領域が少ないことを示す．

4.2 評価法

この実験では，レジストレーションの成功回数，成功時のレジストレーション精度，処理時間を評価に用いる．レジストレーションの精度測定は，文献 9) と同様に，真値または推定値で変換したソース点群の Root Mean Squared Error ($RMSE$) を用いた．$RMSE$ は，式 (8) により求める．

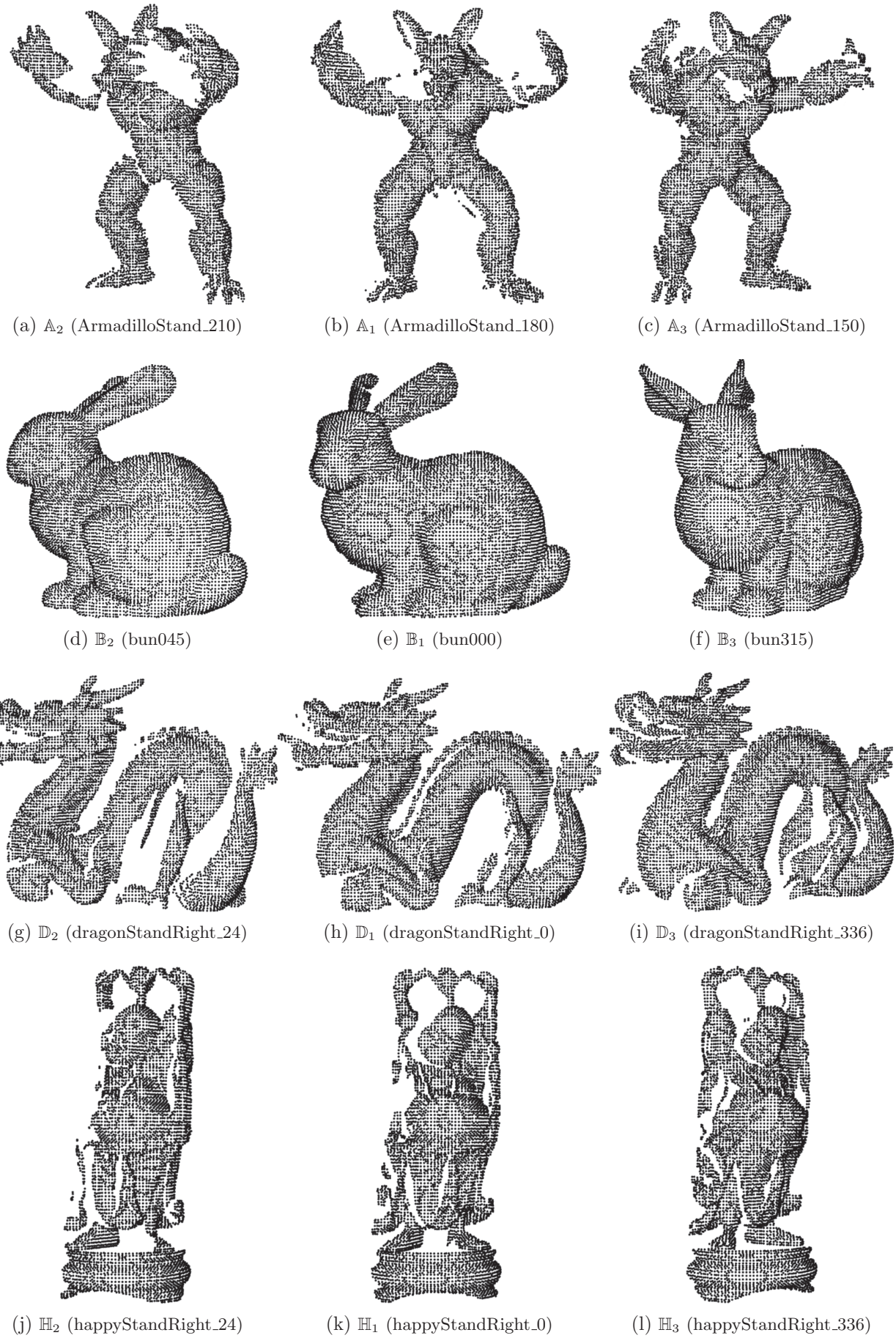
$$RMSE = \sqrt{\frac{\sum_{i=1}^{N_s} ||\boldsymbol{T}_e\, \boldsymbol{p}_i^s - \boldsymbol{T}_{gt}\, \boldsymbol{p}_i^s||^2}{N_s}} \tag{8}$$

ここで，$\boldsymbol{T}_e$ は，ECR で推定した変換行列，$\boldsymbol{T}_{gt}$ は各データセットにおける真の変換行列を示す．$\mathbb{S}$ の解像度よりも $RMSE$ が小さい ($RMSE < 1.0$[mr]) ときを，レジストレーション成功とみなし，このときの $RMSE$ 値をレジストレーション精度とした．処理時間は，成否にかかわらず 30 回の試行の平均値とした．

4.3 実装法とパラメータ

本実験では，KPP の抽出元となるキーポイント検出に ISS を用いた．これは，支持半径 r_{iss} 内の点分布の共分散行列から固有値を求め，これらの割合が閾値 Th_{21} と Th_{32} を超えたもののみキーポイントとして取り出す方式である．突出値は第 3 固有値がそのまま採用される．文献 18) によると，絶対及び相対再現性のバランスがよく，しかも処理時間の短い検出器と評されている．ISS 以外にも，法線推定，境界抽出，最近傍探索は PCL に実装された関数を使用する．PCL において，最近傍探索には FLANN[23] による kd-Tree[24] が採用されており，比較的高速に処理可能である．乱数の生成には，Boost ライブラリで実装されているメルセンヌ・ツイスタ[25] を用いた．また，SADE による ECR は，PCL をベースに我々が実装した．ECR の比較対象としては，PCL に実装されている ICP をそのまま使用した．これは FS の算出に Mean(·) を用いており，KPP による ECR と同様に，オクルージョンの影響を大きく受ける．

各手法のパラメータを**表 2** に示す．“Pre. ex.” は予備実験で得た最適の値である．ISS の r_{nms} は，各点群ごとに 10 個のキーポイントが検出されるように調整した．文献 19) では $Th_{21} = 0.975$ が推奨されているが，この値では平面上からも数多くのキーポイントが検出されたため，本実験では $Th_{21} = 0.6$ として，より突出した位置のキーポイントのみ



(a) $\mathbb{A}_2$ (ArmadilloStand_210) (b) $\mathbb{A}_1$ (ArmadilloStand_180) (c) $\mathbb{A}_3$ (ArmadilloStand_150)

(d) $\mathbb{B}_2$ (bun045) (e) $\mathbb{B}_1$ (bun000) (f) $\mathbb{B}_3$ (bun315)

(g) $\mathbb{D}_2$ (dragonStandRight_24) (h) $\mathbb{D}_1$ (dragonStandRight_0) (i) $\mathbb{D}_3$ (dragonStandRight_336)

(j) $\mathbb{H}_2$ (happyStandRight_24) (k) $\mathbb{H}_1$ (happyStandRight_0) (l) $\mathbb{H}_3$ (happyStandRight_336)

図 7 実験に用いた点群

Fig. 7 Point clouds used in experiments

表 2 実験に用いたパラメータ
Table 2 Parameters used in experiments

Method	Parameter	Value	Unit	Remark
Boundary	r_{ne}	10.0	mr	Pre. ex.
	r_{be}	4.0	mr	Pre. ex.
	Th_{be}	90.0	deg	Ref. 20)
ISS	r_{iss}	10.0	mr	Pre. ex.
	Th_{21}	0.600	-	Pre. ex.
	Th_{32}	0.975	-	Ref. 19)
	r_{nms}	-	mr	Various
	Keypoints	10	-	Fix
KPP	r_{kpp}	4.0	mr	Pre. ex.
	r_{iso}	5.0	mr	Pre. ex.
ECR	Population	30	-	Pre. ex.
	Rotation	±180	deg	Ref. 9)
	Translation	±0.04	m	Ref. 9)
SADE	F_l	0.1	-	Ref. 21)
	F_u	0.9	-	Ref. 21)
	τ_1	0.1	-	Ref. 21)
	τ_2	0.1	-	Ref. 21)
Termination	Diff. FS	1.0e-10	m^2	Pre. ex.
Experiment	Trial times	30	-	Ref. 9)

表 3 実験用計算機の性能
Table 3 Specification of the computer

CPU	Intel Core i5-4570SK (3.50GHz)
Memory	8.0 [GByte]
OS	Ubuntu version 14.04.3 (64bit)
Compiler	Clang++ version 3.4
Program library	PCL version 1.7.2

を検出した．ECR の回転と並進は，表 2 に示す範囲内でランダムに値を取得し，30 個の個体を生成した．終了条件の Diff. FS は，FS が更新された場合，前回との FS の差分がこの値より小さい場合に ECR を終了する．

ICP の初期位置は，ECR における回転と並進と同様の範囲内でランダムに値を取得し，1 度だけソース点群を変換することにより 30 個の初期位置を設定した．また，ICP の終了条件も ECR と同様である．

本実験は，**表 3** に示す計算機で 30 回試行した．

4.4 KPP の性能評価実験

本実験では，KPP の性能を確認した．**表 4** は，ICP，従来の ECR(表中では ECR と表記) 及び KPP 法を用いた ECR(表中では KPP と表記) の試行結果である．Proc. points はソース点群における処理対象点数，すなわち従来方式はソース点群そのものの構成点数 N_s，提案方式は KPP の構成点数 N_k を示した．Success Num. は成功とみなす $RMSE$ にまで収束した回数，Success $RMSE$ は成功時の $RMSE$ の平均値である．処理時間は，Prep. Time が KPP 抽出までに要した処理時間の平均，ECR Time が ECR 開始から終了条件に至るまでの処理時間の平均である．一度も成功に至らなかったデータセットの $RMSE$ には，全体の中央値を括弧内に示した．**図 8** は，KPP によらない ECR の平均収束時間である

表 4 KPP の性能評価実験結果
Table 4 Result of KPP performance

Dataset	Method	Proc. points	Success		Time [sec]	
			Num.	$RMSE$	Prep.	ECR
$\{\mathbb{A}_1, \mathbb{A}_2\}$	ICP	5095	0/30	(13.945)	-	0.354
	ECR	5095	30/30	0.238	-	176.362
	KPP	234	30/30	0.329	0.207	4.865
$\{\mathbb{A}_1, \mathbb{A}_3\}$	ICP	5559	0/30	(12.957)	-	0.498
	ECR	5559	28/30	0.472	-	221.067
	KPP	252	29/30	0.901	0.242	4.761
$\{\mathbb{B}_1, \mathbb{B}_2\}$	ICP	6813	0/30	(9.931)	-	1.116
	ECR	6813	22/30	0.644	-	330.911
	KPP	267	30/30	0.337	0.263	7.303
$\{\mathbb{B}_1, \mathbb{B}_3\}$	ICP	6831	0/30	(9.765)	-	0.853
	ECR	6831	29/30	0.404	-	227.242
	KPP	237	0/30	(4.696)	0.246	4.606
$\{\mathbb{D}_1, \mathbb{D}_2\}$	ICP	6312	10/30	0.352	-	0.702
	ECR	6312	29/30	0.211	-	173.601
	KPP	201	30/30	0.253	0.272	5.022
$\{\mathbb{D}_1, \mathbb{D}_3\}$	ICP	7512	7/30	0.868	-	0.992
	ECR	7512	25/30	0.580	-	275.054
	KPP	198	0/30	(8.354)	0.329	4.727
$\{\mathbb{H}_1, \mathbb{H}_2\}$	ICP	5304	0/30	(8.567)	-	0.643
	ECR	5304	30/30	0.270	-	139.846
	KPP	248	29/30	0.596	0.211	5.426
$\{\mathbb{H}_1, \mathbb{H}_3\}$	ICP	5600	11/30	0.887	-	1.262
	ECR	5600	30/30	0.237	-	130.812
	KPP	228	29/30	0.754	0.206	4.522

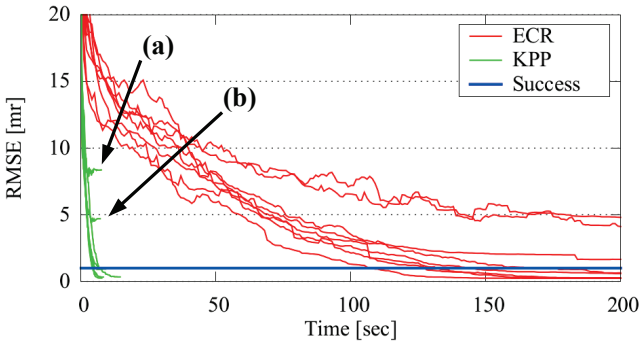


図 8 ECR の収束の様子
Fig. 8 Convergence of ECR

200 秒までの，処理時間と $RMSE$ の平均値の関係を示す．8 個のデータセットにおける収束の様子を，従来方式は赤線，提案方式は緑線で表示した．青線は成否の判定基準 (1.0[mr]) である．

ICP は全ての方式の中で最も高速に収束したが，その成功率は 2 つの ECR を大きく下回り，レジストレーションはほとんど成功しなかった．実際は，最適解付近 ($RMSE \simeq$1.5[mr]) に収束したものも存在するが，これを成功とみなしても全体の成功回数は 10 回程度であった．すなわち，本実験に用いた範囲で回転及び並進した初期位置にあるデータセットの場合，ICP の適用は適切ではないといえる．

従来の ECR は，約 93%の割合で成功しており，その精度も成否判定基準の 1.0[mr] を下回っている場合が多かった．し

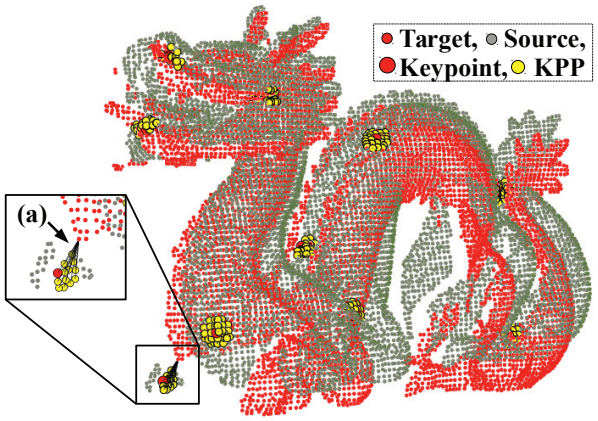


図 9 オクルージョンにある KPP の影響
Fig. 9 Influence of KPP in occlusion

かし，処理時間は約 200[s] であり，文献 10) のとおり膨大な計算量を必要とした．提案方式の KPP 法を用いた ECR は，$\{\mathbb{B}_1, \mathbb{B}_3\}$ と $\{\mathbb{D}_1, \mathbb{D}_3\}$ を除き，約 98%の割合で成功し，その処理時間は 5.1[s] 程度であった．図 8 にも示されるとおり，提案方式の収束までの時間的な優位性は明らかであった．ただし，図 8(a) は $\{\mathbb{D}_1, \mathbb{D}_3\}$，図 8(b) は $\{\mathbb{B}_1, \mathbb{B}_3\}$ の収束結果であり，どちらも成否判定基準には至らずに Diff. FS に関する終了条件を満たした．この 2 つのデータセットは，終了条件を設定せずに処理を継続しても $RMSE$ の変動は 0.001 程度であり，成功に転じることはなかった．

成功時の $RMSE$ を比較すると，$\{\mathbb{B}_1, \mathbb{B}_2\}$ を除き，提案方式のほうが低精度であった．これは，KPP に含まれたオクルージョンの影響であり，N_k が小さいためその影響を低減できなかったためである．$\{\mathbb{B}_1, \mathbb{B}_2\}$ は，全 KPP がオクルージョン領域になかったため，提案方式が最も高精度であった．

$\{\mathbb{B}_1, \mathbb{B}_3\}$ と $\{\mathbb{D}_1, \mathbb{D}_3\}$ のデータセットについては，KPP の一部がオクルージョン領域に抽出されていたことを確認した．**図 9** は，$\{\mathbb{D}_1, \mathbb{D}_3\}$ を用いた試行結果の一例である．赤点は $\mathbb{T}$，灰点は $\mathbb{S}$，大きい赤点はキーポイント，大きい黄点は KPP を示しており，黒線は最近傍点への距離を示す．黒線が短いほど FS が小さくなり，KPP の位置が適正であればこれに伴って $RMSE$ も小さくなる．しかしながら，図 9(a) に示す KPP が $\mathbb{T}$ のオクルージョンの位置にあり，長い黒線が多くあることがわかる．すなわち，ある程度まで最適解に収束した段階で，オクルージョンの影響を排除しきれず局所解に収束したといえる．

以上より，ICP ではレジストレーションが成功できないデータセットであっても，ECR では概ね成功できることを確認した．また，KPP 法を用いた ECR は 2 個のデータセットを除き，従来の ECR と同等の成功率を示し，かつ約 40 倍高速であることを確認した．成功率を考慮すると，KPP を用いた ECR が最も優れたレジストレーション方式であるといえる．ただし，成功時のレジストレーション精度は従来方式よりも多少低く，KPP の位置によっては局所解に至るデータセットもあった．これらの原因は，オクルージョンの影響を強く受けたためである．

表 5 KPP-BR の性能評価実験結果
Table 5 Result of KPP-BR performance

Dataset	Method	Proc. points	Success		Time [sec]	
			Num.	$RMSE$	Prep.	ECR
$\{\mathbb{A}_1, \mathbb{A}_2\}$	KPP	234	30/30	0.329	0.207	4.865
	KPP-BR	268	30/30	0.215	0.072	5.842
$\{\mathbb{A}_1, \mathbb{A}_3\}$	KPP	252	29/30	0.901	0.242	4.761
	KPP-BR	287	28/30	0.473	0.086	5.709
$\{\mathbb{B}_1, \mathbb{B}_2\}$	KPP	267	30/30	0.337	0.263	7.303
	KPP-BR	302	28/30	0.266	0.194	7.055
$\{\mathbb{B}_1, \mathbb{B}_3\}$	KPP	237	0/30	(4.696)	0.246	4.606
	KPP-BR	284	30/30	0.639	0.162	8.665
$\{\mathbb{D}_1, \mathbb{D}_2\}$	KPP	201	30/30	0.253	0.272	5.022
	KPP-BR	269	27/30	0.147	0.101	7.890
$\{\mathbb{D}_1, \mathbb{D}_3\}$	KPP	198	0/30	(8.354)	0.329	4.727
	KPP-BR	284	30/30	0.228	0.133	6.772
$\{\mathbb{H}_1, \mathbb{H}_2\}$	KPP	248	29/30	0.596	0.211	5.426
	KPP-BR	282	28/30	0.437	0.074	7.406
$\{\mathbb{H}_1, \mathbb{H}_3\}$	KPP	228	29/30	0.754	0.206	4.522
	KPP-BR	285	29/30	0.527	0.081	5.843

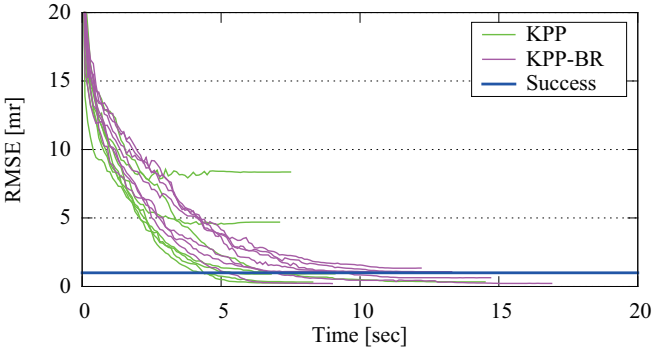


図 10 KPP と KPP-BR の収束の様子
Fig. 10 Convergence of KPP and KPP-BR

4.5 KPP-BR の性能評価実験

本節では，KPP-BR の性能を確認する．KPP を用いた ECR と，KPP-BR を用いた ECR の試行結果を**表 5** に，収束時間と $RMSE$ の関係を**図 10** に示す．

$\{\mathbb{B}_1, \mathbb{B}_3\}$ と $\{\mathbb{D}_1, \mathbb{D}_3\}$ の結果が大きく改善されており，全体でも約 97%の割合でレジストレーションが成功した．これは，図 5(a) と図 5(b) からもわかるように，境界点を除去することによってオクルージョンの位置に KPP を抽出しなかったためといえる．成功時の $RMSE$ は全て改善され，これらの平均値 0.367 は，KPP によらない従来の ECR の平均値 0.382 よりも低かった．図 10 における KPP-BR の結果 (紫線) をみても，成否判定基準である青線を大きく上回るものがないことがわかる．わずかに青線を上回っているものは，失敗時における $RMSE$ の影響を受けた結果である．

3.2.1 項で述べたように，KPP を成す点が KPP-BR では増加しており，ECR の処理時間は平均して約 1.7[s] 増加した．$\{\mathbb{B}_1, \mathbb{B}_2\}$ のとき KPP-BR の方が高速であった理由は，

表 6　KPP 高速化の評価実験結果
Table 6　Effectiveness of acceleration of KPP

(a) Reduction of KPP number

Dataset	Num. KPP	Proc. points	Success Num.	Success $RMSE$	Time [sec] Prep.	Time [sec] ECR
$\{\mathbb{A}_1, \mathbb{A}_2\}$	10	268	30/30	0.215	0.072	5.842
	8	211	27/30	0.251	0.078	4.748
	6	156	29/30	0.309	0.080	3.262
	4	106	10/30	0.234	0.107	2.712
$\{\mathbb{A}_1, \mathbb{A}_3\}$	10	287	28/30	0.473	0.086	5.709
	8	228	24/30	0.473	0.094	4.583
	6	173	29/30	0.575	0.096	3.672
	4	114	21/30	0.628	0.123	2.506
$\{\mathbb{B}_1, \mathbb{B}_2\}$	10	302	28/30	0.266	0.194	7.055
	8	242	27/30	0.361	0.233	6.281
	6	180	28/30	0.441	0.274	5.184
	4	120	9/30	0.558	0.364	4.474
$\{\mathbb{B}_1, \mathbb{B}_3\}$	10	284	30/30	0.639	0.162	8.665
	8	233	30/30	0.594	0.164	6.772
	6	172	30/30	0.704	0.199	4.937
	4	115	9/30	0.785	0.234	3.949
$\{\mathbb{D}_1, \mathbb{D}_2\}$	10	269	27/30	0.147	0.101	7.890
	8	210	26/30	0.110	0.101	5.391
	6	158	25/30	0.134	0.110	4.145
	4	101	16/30	0.322	0.116	3.000
$\{\mathbb{D}_1, \mathbb{D}_3\}$	10	284	30/30	0.228	0.133	6.772
	8	227	28/30	0.263	0.136	5.755
	6	170	22/30	0.285	0.140	4.373
	4	115	14/30	0.684	0.149	3.084
$\{\mathbb{H}_1, \mathbb{H}_2\}$	10	282	28/30	0.437	0.074	7.406
	8	229	17/30	0.453	0.074	5.643
	6	175	3/30	0.439	0.079	3.472
	4	121	5/30	0.605	0.091	2.884
$\{\mathbb{H}_1, \mathbb{H}_3\}$	10	285	29/30	0.527	0.081	5.843
	8	228	25/30	0.556	0.083	4.907
	6	164	11/30	0.875	0.087	3.945
	4	111	0/30	(9.651)	0.090	2.879

(b) Reduction of points constructing KPP

Dataset	Rate [%]	Proc. points	Success Num.	Success $RMSE$	Time [sec] Prep.	Time [sec] ECR
$\{\mathbb{A}_1, \mathbb{A}_2\}$	100	268	30/30	0.215	0.072	5.842
	75	200	30/30	0.251	0.072	4.069
	50	136	30/30	0.349	0.073	3.038
	25	71	28/30	0.410	0.071	1.645
$\{\mathbb{A}_1, \mathbb{A}_3\}$	100	287	28/30	0.473	0.086	5.709
	75	214	28/30	0.464	0.090	4.385
	50	144	26/30	0.490	0.090	2.868
	25	76	24/30	0.482	0.088	1.558
$\{\mathbb{B}_1, \mathbb{B}_2\}$	100	302	28/30	0.266	0.194	7.055
	75	226	30/30	0.307	0.194	5.857
	50	152	30/30	0.355	0.196	4.071
	25	80	29/30	0.387	0.192	2.081
$\{\mathbb{B}_1, \mathbb{B}_3\}$	100	284	30/30	0.639	0.162	8.665
	75	212	30/30	0.639	0.161	6.390
	50	143	30/30	0.629	0.160	4.319
	25	75	30/30	0.654	0.159	2.382
$\{\mathbb{D}_1, \mathbb{D}_2\}$	100	269	27/30	0.147	0.101	7.890
	75	200	29/30	0.117	0.099	5.496
	50	136	29/30	0.151	0.098	3.596
	25	71	28/30	0.206	0.098	2.241
$\{\mathbb{D}_1, \mathbb{D}_3\}$	100	284	30/30	0.228	0.133	6.772
	75	212	30/30	0.230	0.131	4.941
	50	143	28/30	0.271	0.131	3.280
	25	75	28/30	0.299	0.132	1.990
$\{\mathbb{H}_1, \mathbb{H}_2\}$	100	282	28/30	0.437	0.074	7.406
	75	210	25/30	0.380	0.075	5.242
	50	142	25/30	0.427	0.073	3.636
	25	74	23/30	0.436	0.073	1.802
$\{\mathbb{H}_1, \mathbb{H}_3\}$	100	285	29/30	0.527	0.081	5.843
	75	213	26/30	0.528	0.082	4.222
	50	144	26/30	0.532	0.080	3.073
	25	75	25/30	0.570	0.081	1.694

収束に至るまでの世代数が少なかったためである．KPP-BR の前処理時間がわずかに短い理由は，境界点を除去したため ISS 検出に要した時間が削減されたためである．

以上より，KPP を用いた ECR には，境界点の除去が必須といえる．KPP によらない従来の ECR の成功率と精度と比較しても，KPP-BR を用いた ECR が優れており，処理速度は約 30 倍高速である．すなわち，KPP-BR による ECR が最も効果的であることが確認できた．

4.6　KPP の個数と KPP を構成する点数の評価実験

本節では，KPP-BR の抽出結果を基準に，KPP の個数と，KPP を構成する点数をそれぞれ削減してその効果を確認する．個数削減については，NMS 半径を手動で微調整し，$\{10, 8, 6, 4\}$ 個の KPP を抽出した．点数削減については，$\{100, 75, 50, 25\}$[%] の割合でランダムにリサンプリングした．数量削減の結果は**表 6** (a) に，点数削減の結果は表 6 (b) に示す．

表 6 (a) をみると，数量削減により処理時間を削減できているが，成功率と成功時の $RMSE$ は悪化した．特に，$\{\mathbb{H}_1, \mathbb{H}_2\}$ と $\{\mathbb{H}_1, \mathbb{H}_3\}$ において，KPP が 6 個以下のときの性能低下は顕著である．図 7 より，これらのデータセットが他と比較して突出した部分が少なく，円柱の表面をセンシングしたような形状となっているためと考えられる．点群の形状に適した KPP の数量と位置は異なるといえるが，事前にこれを決定することは困難である．すなわち，KPP の個数削減による高速化は，成功率とのトレードオフになるといえる．

点数削減の結果は，成功率と精度を概ね保ちつつ，処理時間を向上できるため，有効であることを確認できた．サンプリング率 100%と 25%を比較すると，成功率は 96%から 90%，精度は 0.367 から 0.411 程度の悪化で，処理速度は約 3.5 倍改善できた．

以上より，数量削減及び点数削減はともに，高速化の効果があることが確認できた．数量削減は，成功率と処理時間のトレードオフであるが，点数削減は成功率と成功時の精度を極端に落とすことなく処理時間を短縮できた．特に，点数削減の 25%リサンプリングでは，リサンプリング無しよりも約

3.5 倍高速であった．これは，KPP によらない従来の ECR と比較すると，約 100 倍高速である．点数の削減が効果的であるのは，KPP を用いた ECR の精度が，KPP の位置に強く依存しているためである．

4.7 KPP 法のノイズ耐性評価実験

本実験は，前項で性能を確認した KPP-BR と点数削減の結果を基準に，ノイズを付加したデータセットを用いることで，KPP 法のノイズ耐性を確認した．データセットには，ターゲット点群及びソース点群の両方に対し，試行の度に異なるシードで 0.5[mr] のレベルのガウシアンノイズを付加した．0.5[mr] レベルは，文献 18) において最も強いレベルである．その他のパラメータは，前項と全く同じものを使用した．また，本実験ではノイズによりレジストレーションの成功数低下が予想されるため，全試行の $RMSE$ の中央値も確認し，その耐性を評価することとした．

この実験結果を，**表 7** に示す．各データセットの第 1 行目は，ノイズのないデータセットの結果を示しており，表 6(b) と同値である．表中の#KPP は，抽出された KPP の個数の平均値である．KPP の個数が変動した理由は，ノイズによる形状の変化のためである．特に Bunny($\mathbb{B}$) では，滑らかな表面がノイズによって突出した部分となり，この位置にキーポイントが検出されたため，KPP の個数の増加が顕著であった．Median $RMSE$ は，全試行の $RMSE$ の中央値を示す．

サンプリング率が 100%(KPP 点数削減の適用なし) のとき同士を比較したとき，全体的に $RMSE$ の中央値，成功回数と成功時の $RMSE$ が悪化したことが確認できる．$RMSE$ の中央値に着目すると，ノイズのある場合は全体で 0.528 程度の増加が確認された．ただし，$RMSE$ の中央値は 1.0 付近で収束しており，全データセットで最適解に近い解を得られているといえる．成功数に着目すると，$\{\mathbb{A}_1, \mathbb{A}_2\}$，$\{\mathbb{B}_1, \mathbb{B}_2\}$ 及び $\{\mathbb{B}_1, \mathbb{B}_3\}$ はノイズのある場合，成功率が約 98%から 44%程度にまで大きく低下していた．これらの $RMSE$ の中央値が，成功とみなす閾値の 1.0 をわずかに超えており，これに伴って成功回数も半減したと考えられる．また，$\{\mathbb{A}_1, \mathbb{A}_3\}$ のサンプリング率 25%と，$\{\mathbb{H}_1, \mathbb{H}_2\}$ のサンプリング率 75%以下の場合の成功率低下も，同様の要因といえる．一方で $\{\mathbb{D}_1, \mathbb{D}_2\}$，$\{\mathbb{D}_1, \mathbb{D}_3\}$ 及び $\{\mathbb{H}_1, \mathbb{H}_3\}$ は，サンプリング率を低下させても成功率を維持していた．これは，KPP の個数と位置がノイズのない場合と同様であったためであり，結果が同じ傾向を示したものと考えられる．処理時間については，点数削減の割合に比例して改善されており，前項の実験と同様の傾向を確認した．

以上から，ノイズのある場合においても，$RMSE$ の中央値は成功とみなす精度の解付近にまで収束しており，そのノイズ耐性を確認できた．また，点数削減による効果はノイズのない場合と同様に，成功率と成功時の精度を極端に落とすことなく処理時間を短縮できている．

表 7 ノイズ耐性の評価実験結果
Table 7 Result of noise robustness of KPP

Dataset (#KPP)	Rate [%]	Median $RMSE$	Success Num.	Success $RMSE$	Time [sec] Prep.	Time [sec] ECR
$\{\mathbb{A}_1, \mathbb{A}_2\}$ (11.2)	100	0.210	30/30	0.215	0.072	5.842
	100	1.110	12/30	0.632	0.069	5.149
	75	1.200	10/30	0.580	0.069	3.434
	50	1.320	10/30	0.793	0.068	2.296
	25	1.213	9/30	0.829	0.069	1.657
$\{\mathbb{A}_1, \mathbb{A}_3\}$ (11.1)	100	0.468	28/30	0.473	0.086	5.709
	100	0.931	19/30	0.779	0.081	4.833
	75	0.891	21/30	0.776	0.078	3.901
	50	0.912	19/30	0.769	0.081	2.689
	25	1.047	12/30	0.748	0.078	1.447
$\{\mathbb{B}_1, \mathbb{B}_2\}$ (16.4)	100	0.265	28/30	0.266	0.194	7.055
	100	1.042	14/30	0.597	0.162	9.749
	75	1.056	14/30	0.662	0.165	7.801
	50	1.044	14/30	0.614	0.165	4.932
	25	1.133	12/30	0.689	0.163	2.922
$\{\mathbb{B}_1, \mathbb{B}_3\}$ (21.0)	100	0.645	30/30	0.639	0.161	8.665
	100	1.031	14/30	0.768	0.135	11.399
	75	1.020	14/30	0.810	0.136	8.059
	50	1.050	12/30	0.772	0.135	6.446
	25	1.078	12/30	0.731	0.135	3.382
$\{\mathbb{D}_1, \mathbb{D}_2\}$ (9.9)	100	0.113	27/30	0.147	0.101	7.890
	100	0.658	27/30	0.594	0.089	5.119
	75	0.738	23/30	0.666	0.089	4.350
	50	0.727	19/30	0.605	0.090	2.655
	25	0.855	20/30	0.723	0.090	1.454
$\{\mathbb{D}_1, \mathbb{D}_3\}$ (11.4)	100	0.221	30/30	0.228	0.133	6.772
	100	0.684	26/30	0.667	0.120	5.522
	75	0.685	25/30	0.667	0.118	3.674
	50	0.730	21/30	0.634	0.118	2.554
	25	0.853	21/30	0.687	0.118	1.358
$\{\mathbb{H}_1, \mathbb{H}_2\}$ (13.7)	100	0.450	28/30	0.437	0.074	7.406
	100	0.997	25/30	0.990	0.065	5.042
	75	1.088	12/30	0.811	0.063	4.189
	50	1.067	13/30	0.809	0.064	2.805
	25	1.155	9/30	0.886	0.064	1.625
$\{\mathbb{H}_1, \mathbb{H}_3\}$ (10.1)	100	0.525	29/30	0.527	0.081	5.843
	100	0.671	25/30	0.585	0.070	4.192
	75	0.706	24/30	0.607	0.068	2.720
	50	0.709	23/30	0.603	0.070	2.020
	25	0.850	20/30	0.718	0.068	1.115

5. むすび

本稿では，ECR の初期位置に依存せずに高精度な結果を得られるという特長を保持しつつ，処理コストを低減するための KPP 法を提案した．KPP は，キーポイントの周囲をパッチ状に抽出した点群である．KPP 法は ECR の最近傍探索回数を大きく削減でき，処理速度の向上に有用である．さらに，境界点除去によるオクルージョンの影響の低減，NMS 処理による数量の削減，リサンプリングによる点数の削減といった改善手法も提案した．

8 組のデータセットを用いた評価実験により，KPP を用いた ECR は ICP のように初期位置の影響を受けず，KPP によらない従来の ECR と比較して約 40 倍高速化できることを確認した．また，境界点を除去する KPP-BR は，オク

ルージョンを多く含むデータセットであっても，従来の ECR と同等以上の成功率と精度を示した．処理時間の削減については，数量削減手法は成功率とのトレードオフである一方，点数削減は成功率と精度を維持できることを確認した．点数削減にて，元の 25%まで点数を削減してもレジストレーション精度に与える影響は小さかったことから，ECR の性能は KPP の抽出位置に依存するといえる．したがって，KPP は境界点除去の併用が必須であり，更に高速化が必要な場合には点数削減の適用が適切である．これら提案方式の組み合わせは，KPP によらない従来の ECR と同等以上の成功率と精度を示しつつ，最大で約 100 倍以上の高速化を実現した．また，ノイズを含む点群の場合でも，その成功率と精度を維持しつつ，レジストレーションの高速化に貢献することを確認した．

参 考 文 献

1) H. Wei, M. Bartels, 3D Digital Elevation Model Generation, 3D Imaging, Analysis and Applications, Springer, London, pp.367–416 (2012).
2) A. Mian, N. Pears, 3D Face Recognition, 3D Imaging, Analysis and Applications, Springer, London, pp.311–366 (2012).
3) W.A.P. Smith, Representing, Storing and Visualizing 3D Data, 3D Imaging, Analysis and Applications, Springer, London, pp.139–182 (2012).
4) K. Reinhard, P. Nick, L. Yonghuai, Introduction, 3D Imaging, Analysis and Applications, Springer, London, pp.1–34 (2012).
5) The Stanford 3D Scanning Repository, http://graphics.stanford.edu/data/3Dscanrep/ (2018).
6) G. Turk, M. Levoy: “Zippered Polygon Meshes from Range Images”, Proc. of ACM the 21st Annual Conference on Computer Graphics and Interactive Techniques, pp.311–318 (1994).
7) P. Besl, H. McKay: “A Method for Registration of 3-D Shapes”, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.14, No.2, pp.239–256 (1992).
8) A.M. Bronstein, M.M. Bronstein,, M. Ovsjanikov, Feature-Based Methods in 3D Shape Analysis, 3D Imaging, Analysis and Applications, Springer, London, pp.185–219 (2012).
9) J. Santamaría, O. Cordón, S. Damas: “A Comparative Study of State-of-the-art Evolutionary Image Registration Methods for 3D Modeling”, Computer Vision and Image Understanding, Vol.115, No.9, pp.1340–1354 (2011).
10) J. Salvi, C. Matabosch, D. Fofi,, J. Forest: “A Review of Recent Range Image Registration Methods with Accuracy Evaluation”, Image and Vision Computing, Vol.25, No.5, pp.578–596 (2007).
11) U. Castellani, A. Bartoli: “3D Shape Registration”, 3D Imaging, Analysis and Applications, Springer, London, pp.367–416 (2012).
12) K. Uenishi, M. Iwakiri, K. Tanaka: “Efficient Point Clouds Registration Based on Differential Evolution with Keypoint Patches”, Proc. of International Workshop on Image Electronics and Visual Computing, 1C-1 (2014).
13) K. Uenishi, J. Sandoval, M. Iwakiri,, K. Tanaka: “Improved Keypoint Patches in Evolutionary Point Clouds Registration”, Proc. of International Workshop on Image Electronics and Visual Computing, 1C-3 (2017).
14) M. Rodrigues, R. Fisher, Y. Liu: “Special Issue on Registration and Fusion of Range Images”, Computer Vision and Image Understanding, Vol.87, No.1, pp.1–7 (2002).
15) C.K. Chow, H.T. Tsui, T. Lee: “Surface Registration Using a Dynamic Genetic Algorithm”, Pattern Recognition, Vol.37, No.1, pp.105–117 (2004).
16) R. Storn, K. Price: “Differential Evolution– A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces”, Journal of global optimization, Vol.11, No.4, pp.341–359 (1997).
17) K. Uenishi, M. Iwakiri: “Virtual Feature Point Extraction from Polyhedral Structure”, Proc. of IEEE International Symposium on Intelligent Signal Processing and Communication Systems, pp.519–524 (2013).
18) F. Tombari, S. Salti, L. Di Stefano: “Performance Evaluation of 3D Keypoint Detectors”, International Journal of Computer Vision, Vol.102, No.1-3, pp.198–220 (2013).
19) Y. Zhong: “Intrinsic Shape Signatures: A Shape Descriptor for 3D Object Recognition”, Proc. of International Conference on Computer Vision Workshops, pp.689–696 (2009).
20) R.B. Rusu: “Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments”, Ph.D. thesis, Computer Science department, Technische Universitaet Muenchen (2009).
21) J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer: “Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems”, IEEE Trans. on Evolutionary Computation, Vol.10, No.6, pp.646–657 (2006).
22) R.B. Rusu, S. Cousins: “3D is here: Point Cloud Library (PCL)”, Proc. of IEEE International Conference on Robotics and Automation, pp.1–4 (2011).
23) M. Muja, D.G. Lowe: “Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration”, Proc. of International Conference on Computer Vision Theory and Application, pp.331–340 (2009).
24) D.A. Simon: “Fast and Accurate Shape-Based Registration”, Ph.D. thesis, Citeseer (1996).
25) M. Matsumoto, T. Nishimura: “Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator”, ACM Trans. on Modeling and Computer Simulation, Vol.8, No.1, pp.3–30 (1998).

(2017 年 10 月 15 日 受付)
(2018 年 2 月 10 日 再受付)

植 西 一 馬 （正会員）
2008 年 防衛大学校情報工学科卒業．2014 年 防衛大学校理工学研究科前期課程情報数理専攻修了．現在，信州大学大学院総合工学系研究科博士課程システム開発工学専攻在学中．修士 (工学)．3 次元形状情報処理に関する研究に従事．

サンドバル・ハイメ （学生会員）
2009 年 メキシコ・El Fuerte Valley University 卒業．2017 年 信州大学信州大学理工学系研究科修士課程電気電子システム工学専攻修了．現在，信州大学大学院総合工学系研究科博士課程システム開発工学専攻在学中．修士 (工学)．3 次元情報処理，コンピュータビジョン，画像処理に関する研究に従事．

岩 切 宗 利 （正会員）
1993 年 防衛大学校情報工学科卒業．1998 年 防衛大学校理工学研究科情報数理専攻修了．1999 年 防衛大学校情報工学科助手．2005 年 同講師．2015 年 同准教授．博士 (工学)．マルチメディアと情報セキュリティに関する研究に従事．情報処理学会会員．

田 中 清 （フェロー）
1984 年 防衛大学校電気工学科卒業．1989 年 防衛大学校理工学研究科オペレーションズリサーチ専攻修了．1992 年 防衛大学校情報工学科助手．1995 年 信州大学工学部電気電子工学科助教授．2006 年 同教授．2015 年 信州大学副学長，現在に至る．博士 (工学)．画像・映像処理，色覚，三次元点群処理，情報ハイディング，進化計算，多目的最適化，知的電力網・交通システムなどの研究に従事．IEEE，電子情報通信学会，情報処理学会，日本眼光学学会，日本産業・労働・交通眼科学会などの会員．本学会前編集委員長．