選択的統合処理に基づく CIF ベース大域的自己位置推定

脇 田 翔 平^{*1} 中 村 恭 之^{*2}

CIF-based Global Scan Matching Method Enhanced by Selective Merging 2D Range Scans

Shohei Wakita^{*1} and Takayuki Nakamura^{*2}

This paper proposes a selective merging method of 2D range scans for enhancing our CIF-based scan matching algorithm which is a global scan matching algorithm using the CIF descriptors and a geometric constraint between keypoints. The selective merging method can generate a local map based on “terrain complexity” without fixing the number of scans to use. Our enhanced version of the CIF-based scan matching algorithm can perform global scan matching more robustly in a large cluttered environments without using an initial alignment. Through experiment in real environment, we confirm the validity of our method.

Key Words: Selective Merging, Terrain Complexity, CIF Descriptor, Global Scan Matching

1. 緒 言

自律型移動ロボット普及のためには、自動的な環境地図構築技術や大域的自己位置推定技術が必要不可欠である。大域的自己位置推定技術では、ロボットの初期の位置姿勢を与えずに、ロボットの地図内における位置姿勢を推定しなければならない。さらに、地図作成時と環境が変化した場合、意図せずに自分の位置を見失った場合や別の場所に移動された場合などに、正確に大域的自己位置推定できる能力を持つことが望まれる。

二次元レーザスキャナを用いて大域的自己位置推定を実現するために、地図データ内の対応する個所とスキャン（二次元レーザスキャナで取得した 1 時刻のデータ）の位置合わせを行うスキャンマッチング法がよく用いられる。これまでに数多くのスキャンマッチング法が提案されてきた[1][2]。しかし、スキャンマッチング法には、位置合わせが困難になる状況が存在する（スキャンにノイズやオクルージョンが発生している場合、環境中に移動物体が存在する場合、スキャンの位置合わせにおいて有用な形状特徴が不足している場合など）。これらの問題を解決する方法として、複数時刻のスキャンを用いて局所地図を生成し、生成した局所地図と環境地図とでスキャンマッチングを行い、自己位置推定する方法がある。この場合は、連続して入力されるスキャンの中から局所地図生成に用いるスキャンをどの

ように選択するかが問題となる。局所地図生成に用いるスキャンの数をあらかじめ限定すると、生成された局所地図内に環境を表す形状特徴が不足する可能性があり、そのような状況では、正しく位置合わせを行うことが難しくなる。一方で、一時刻分のスキャンに形状特徴が豊富にある場合、複数時刻分のスキャンを用いて局所地図を生成する必要はない。大域的自己位置推定に用いるスキャン数を最小にするためには、環境内に含まれる形状特徴に応じて局所地図を生成するか否かを判断して、局所地図生成に用いるスキャンを適切に選択して位置合わせすることが望ましい。

そこで、本研究では、スキャンの形状特徴を表す特徴量“地形複雑度 (terrain complexity)”を提案し、この特徴量に基づいて局所地図を作成する方法“選択的統合処理”を提案する。“地形複雑度”は、スキャン内の環境の形状特徴を表し、それが十分に含まれているか否か（環境の形状特徴の有無）を判断することを、2 クラスの識別問題として捉え、何らかの識別器によってその判断を下す。“選択的統合処理”は、識別器の判断結果に基づき、用いるスキャンを適切に選択して局所地図を生成する手法である。この選択的統合処理により、用いるスキャンの数をあらかじめ決定することなく、形状特徴が十分に含まれた局所地図を生成できる。この局所地図を用いることで、我々の研究グループで提案した CIF-based スキャンマッチング法[3]~[5]での大域的自己位置推定法がより頑健に行えるようになる。提案手法の有効性を検証するために、実環境の地図を用いて大域的自己位置推定を行う実験を行い、地形複雑度に基づいてスキャンに含まれる形状特徴が十分か否かを判断できることを示し、位置合わせに用いるスキャンの数を限定せずに局所地図を

原稿受付 2016 年 12 月 12 日

^{*1}和歌山大学大学院システム工学研究科

^{*2}和歌山大学システム工学部

^{*1}Faculty of Systems Engineering, Wakayama University

^{*2}Wakayama University

■ 本論文は新規性（要素分野）で評価されました。

生成でき、その局所地図を用いて大域的自己位置を頑健に行えることを示す。また、いわゆる「ロボット誘拐問題」の状況を想定した Monte Carlo Localization (MCL) 法 [6] を用いた大域的自己位置推定の結果と比較して、提案手法の有効性を示す。

2. 関 連 研 究

大域的自己位置推定法は、位置姿勢の初期値を持たせず、ロボットの持つ外界センサから得られる情報と環境地図とを照合し、ロボットの位置姿勢を推定する。大域的自己位置推定法に関する研究例では、カメラやレーザスキャナといった外界センサが用いられることが多い。

カメラを用いて大域的自己位置推定を行う研究例 [7] では、環境中のある場所で観測された画像から SIFT 特徴の集合を抽出し、それらのある場所を表す情報として登録し、環境内でこのような登録を多数行いデータベースを作成する。データベース中のこれらの登録情報に関して、相互情報量に基づいて重みづけして大域的自己位置推定を頑健に行うことを試みている。

阪東 [8] らは、二次元測域センサデータの空間周波数解析に基づく大域的自己位置推定法を提案している。この手法では、二次元測域センサから入力されるスキャンデータと、あらかじめ作成されている地図データを、いったん、二次元の画像情報へ変換して、その変換されたデータに対して二次元 FFT により周波数解析することで、入力スキャンと地図との照合を行う。この手法では、その計算過程でデータを量子化する処理が度々行われるため、それらの処理過程における量子化間隔によって、推定される位置姿勢の精度の良し悪しが左右される。また、この手法では、スキャンデータ全体を用いて処理が行われるため、基本的に環境は静的であることが前提である。

Konolige ら [9] は、ロボットの行動範囲内すべての位置姿勢を一定間隔で量子化しておき、時々刻々のスキャンデータと地図との相互相関に基づく尤度を再計算して、環境地図内で自己位置を絞り込んで自己位置推定を行う Grid Localization を提案した。この手法は計算量が多く処理時間が長くなるという問題があったため、Olson ら [10] は、並列計算機の利用や位置姿勢の量子化を階層化し、高速化を図った。Dellaert ら [6] は、Grid Localization と違い、環境地図を格子に区切ることなく、位置姿勢の候補を粒子（パーティクル）で表し、スキャンデータが入力されるたびに尤度の低い粒子を淘汰していくことで自己位置推定を行う Monte Carlo Localization (MCL) 法を提案している。また、MCL 法においていわゆる「ロボット誘拐問題」が発生した場合の解消法として、Gutmann ら [11] は、誘拐発生時にパーティクルの分布を再計算し復帰を試みる方法を提案している。この方法では誘拐による移動先が近距離である場合にもパーティクルの分布を広げすぎってしまうため無駄が大きい。それに対して上田ら [12] は、誘拐発生直前のパーティクルの情報を残しつつパーティクルの分布を広げて、復帰を試みる方法を提案している。

レーザスキャナから得られるスキャンの同士の位置合わせを

行う手法をスキャンマッチングという。スキャンマッチング法を利用した大域的自己位置推定法として、スキャンから抽出した直線分と地図の直線分とを対応づけて自己位置推定を行うもの [1] や、スキャンからコーナ点やステップエッジ点やドアなどの特異な特徴点をスキャン点列から抽出し、それらの特徴点をノードとした完全グラフを構築して、各特徴点のマッチング結果を集計してスキャン間の大域的スキャンマッチングを行う方法 [2] などが存在する。

しかし、スキャンマッチング法は、どんな状況でも正しく位置合わせができるわけではない。スキャンの位置合わせが困難になる代表的な状況として、スキャンにノイズやオクルージョンが発生している場合、環境中に移動物体が存在する場合、スキャンの位置合わせにおいて有用な形状特徴が不足している場合などがあり、これらは同時に発生する可能性もある。これらの問題に対して、前処理によりノイズを除去するもの [13] や、オクルージョンによる特徴点の誤検出に対応するため、スキャン内の各スキャン点において、スキャン全体を表す特徴ベクトルを生成するもの [14] が提案されている。さらに、移動物体が存在する環境に対応するための手法として、移動物体に相当するスキャン内のデータを除去するもの [15] が提案されている。スキャンにその環境を表す形状特徴が不足している場合に対する、一つの解決手法として、原ら [16] は、現在のスキャンのみを用いて位置合わせを行うのではなく、現在から過去の数時刻分のスキャンを用いて局所地図を作成し、その局所地図と地図データとの位置合わせを行う手法を提案している。この手法では、過去 K 時刻分のスキャンを張り合わせて局所地図を生成している。しかし、局所地図生成に用いるスキャンの数をあらかじめ決定しておくと、その局所地図内に環境を表す形状特徴が不足している場合には、局所地図を用いても位置合わせが困難になる。

これまで我々の研究グループでも、スキャンマッチングの一手法として、スキャン内のデータの欠損や変化があっても環境中から不変な箇所を探索し安定して地図を更新できるように、合同変換に不変な CIF 特徴量 (Congruence transformation Invariant Feature) [17] を用いたスキャンマッチング法 [3] [4] (以後、CIF-based スキャンマッチング法とよぶ) を提案した。CIF 特徴量はキーポイント[†]の周囲の点をヒストグラム投票することで、キーポイント周辺の形状を表す特徴量を得る。また、オクルージョンのある環境や形状特徴の乏しい環境内でも大域的自己位置推定を成功させるために、現在と過去 (二時刻分) のスキャンの点列の順序を保存して統合したスキャン (局所地図と同じ) を生成し、このスキャンと地図とを入力として CIF-based スキャンマッチング法により大域的自己位置推定を行う手法を提案した [5]。この手法により、実環境の地図データに対しても、より頑健な位置合わせが可能となった。しかし、局所地図生成に用いるスキャン数をあらかじめ限定してしまうと、それらのスキャンの中に環境の形状特徴が含まれていない可能性があるため、望ましくない。局所地図生成に用いるスキャンの数は、観測した環境に合わせて変化させることが望まれる。そこで、本研究では、自己位置推定に用いるスキャン数を環境の形状によって決定するために、地形情報を数値化する“地形複雑度”とそれ

[†]連続するスキャン点を結んで得られる折れ線において角度差が急激に変化する点をキーポイントとして抽出する。抽出されたキーポイントにおいて、特徴記述を求める。詳細は文献 [3] [17] を参照のこと。

を用いて局所地図を作成する“選択的統合処理”を提案し、その処理を利用した大域的自己位置推定を行う方法についても提案する。1時刻のスキャンに環境の形状特徴が十分に含まれているか否かを判断することを2クラスの識別問題として捉え、識別器によって環境の形状特徴の有無を判断する。この2クラスの識別問題を扱うために、本研究では、Support Vector Machine (SVM) [18] [19] を用いた識別器を利用する。スキャンの地形複雑度と形状特徴の有無を表すラベルからなる訓練データを生成して、SVM を用いた識別器を訓練する。選択的統合処理では、訓練した識別器にスキャンから時々刻々得られる地形複雑度を入力し、スキャンに含まれる形状特徴が十分か否かを判断して、不十分であれば現在のスキャンと過去のスキャンを統合して局所地図を生成する。生成された局所地図に形状特徴が十分含まれているか否かを、同じ識別器を使って判断する。生成される局所地図に形状特徴が十分に含まれていると判定されるまで、統合処理を繰り返し、形状特徴が十分に含まれた局所地図を生成する。この選択的統合処理により、局所地図作成に用いるスキャンの数をあらかじめ決定する必要がなく、また、形状特徴が十分に含まれた局所地図を用いて地図との位置合わせを頑健に行えるようになる。

3. 提案手法

提案手法は、CIF 特徴量 [17] を使用することを前提としている。入力として、二次元レーザスキャンを用いて、ある時刻 t において取得したスキャン $^{raw}\mathbf{P}^t = \{^{raw}\mathbf{p}_1^t, \dots, ^{raw}\mathbf{p}_{N^t}^t\}$ 、環境地図 $^{Map}\mathbf{P}$ 、環境地図内において計算したキーポイントの集合 $^{Map}\mathbf{K}$ 、時刻 $t-1$ までに取得したスキャンとキーポイントの組の集合 $\mathbf{V}^{t-1}$ を用いる。出力は $^{Map}\mathbf{P}$ に対する $^{raw}\mathbf{P}^t$ の位置姿勢 $^{global}\mathbf{q} = (\Delta\tilde{x}, \Delta\tilde{y}, \Delta\tilde{\theta})$ である。ここで、 ^{raw}N は二次元レーザスキャンによって取得したスキャンを構成する点 $^{raw}\mathbf{p}_i^t$ (スキャン点とよぶ) の総数である。 $\mathbf{V}^{t-1}$ は時刻 $t-1$ までに観測されたスキャンと各スキャンにおけるキーポイントの集合の組 $(\mathbf{P}^1, \mathbf{K}^1) \sim (\mathbf{P}^{t-1}, \mathbf{K}^{t-1})$ を保持している LILO 型のデータ構造である。

Fig. 1 は、提案手法の処理手順を表している。この図に示しているように提案手法は ① 前処理+地形複雑度の算出、② 選択的統合処理、③ 位置姿勢推定、という三つのステップからなる。以下では、これら三つのステップについて説明する。

3.1 前処理と地形複雑度の算出

前処理では、二次元レーザスキャンで取得したスキャン $^{raw}\mathbf{P}^t$ と集合 $\mathbf{V}^{t-1}$ を入力して、スキャン $^{raw}\mathbf{P}^t$ を整形し、整形後のスキャンからキーポイント・不連続点を抽出して、整形後のスキャンとキーポイントの組を集合 $\mathbf{V}^{t-1}$ に追加して、整形後のスキャン、キーポイント・不連続点の集合、集合 $\mathbf{V}^t$ を出力する。

整形処理では、スキャン $^{raw}\mathbf{P}^t$ を構成するスキャン点の間隔を均等にする処理を行う。これにより、点間の距離が一定になるため、後に計算する CIF 特徴量の一意性が増す。ここで、

[†]本論文では、連続する2点間の距離が0.5[m]以上離れた場合、その2点を不連続点とする。不連続点では、一意性のある特徴がとれないため、キーポイントとして抽出しない。

Fig. 1 Overview our method

整形後のスキャン $\mathbf{P}^t = \{\mathbf{p}_1^t, \dots, \mathbf{p}_{N^t}^t\}$ とする。整形後のスキャン $\mathbf{P}^t$ から、不連続点[†]を抽出し、抽出された不連続点をまとめて、集合を $\mathbf{B}^t$ として記録する。不連続点は文献 [20] で提案されている手法を利用して抽出する。そして、スキャン $\mathbf{P}^t$ からキーポイントを抽出し、抽出されたキーポイントをまとめて集合 $\mathbf{K}^t$ として記録して、整形後のスキャンとキーポイントの組 $(\mathbf{P}^t, \mathbf{K}^t)$ を集合 $\mathbf{V}^{t-1}$ に追記する。

次に、地形複雑度の算出方法について述べる。

地形複雑度は、整形後のスキャン $\mathbf{P}^t$ 、キーポイントの集合 $\mathbf{K}^t$ 、不連続点の集合 $\mathbf{B}^t$ を用いて算出する。

形状が同じでスケールの異なる二つのスキャンの地形複雑度が同じ値になるように (例えば“幅3[m]のまっすぐな廊下”と“幅1[m]のまっすぐな廊下”の地形複雑度が同じ値になるように)、スキャン $\mathbf{P}^t$ を構成するスキャン点を結んで得られる多角形の面積 (スキャンの面積とよぶ) を計算し、その面積が一定の値 cS になるようにスキャン点の座標を変換する。**Fig. 2** は、スキャン $\{(x_1^t, y_1^t), \dots, (x_{N^t}^t, y_{N^t}^t)\}$ と面積 S^t の関係を表している。

スキャン $\mathbf{P}^t = \{(x_1^t, y_1^t), \dots, (x_{N^t}^t, y_{N^t}^t)\}$ の面積 S^t は、次式のように計算する。

$$S^t = \frac{1}{2} \left| \sum_{i=1}^{N^t+1} (x_i^t y_{i+1}^t - x_{i+1}^t y_i^t) \right| \quad (1)$$

ここで、ロボットの位置を表す座標として $(x_{N^t+1}^t, y_{N^t+1}^t) = (0, 0)$ と $(x_{N^t+2}^t, y_{N^t+2}^t) = (x_1^t, y_1^t)$ を、この計算の都合上スキャン点として追加して、面積の計算に用いる。こうして求められたスキャンの面積が一定の値 cS になるように、次式に従ってスキャン点の座標を変換し、変換後のスキャン $^{trans}\mathbf{P}^t$ を求める。

$$^{trans}\mathbf{P}^t = \left\{ \left(\sqrt{\frac{^cS}{S^t}} x_1^t, \sqrt{\frac{^cS}{S^t}} y_1^t \right), \dots, \left(\sqrt{\frac{^cS}{S^t}} x_{N^t}^t, \sqrt{\frac{^cS}{S^t}} y_{N^t}^t \right) \right\} \quad (2)$$

Fig. 2 How to define the area S^t

この変換後のスキャン $trans P^t$ を用いてスキャン $raw P^t$ の地形複雑度 $raw C^t$ を計算する。地形複雑度を m 次元ベクトル $raw C^t = (raw C_1^t, raw C_2^t, \dots, raw C_m^t)$ で表すものとし、スキャン $trans P^t$ を文献 [20] に従ってセグメンテーションした後、各次元の値は次のように決める[†]。

- $raw C_1^t$ スキャン $trans P^t$ の総点数
- $raw C_2^t$ キーポイント K^t の総点数
- $raw C_3^t$ 不連続点 B^t の総点数
- $raw C_4^t$ スキャン $trans P^t$ 内のセグメントの総数
- $raw C_5^t$ セグメントの長さの平均値
- $raw C_6^t$ セグメントの長さの分散
- $raw C_7^t$ セグメント間の距離の平均値
- $raw C_8^t$ セグメント間の距離の分散
- $raw C_9^t$ スキャン $trans P^t$ の周囲長

スキャンの形状特徴を表すために、総点数やセグメントの情報、周囲長を用いている。提案手法は CIF 特徴量を使用することを前提としているため、キーポイントや不連続点の情報 ($raw C_2^t$, $raw C_3^t$) も特徴量として用いる。CIF 特徴量において、キーポイントはスキャン内にあるコーナの数を表しており、これが豊富にあると地図とスキャンとの対応点の候補が多く得られるため、正しい対応点を得られる可能性が高まる。不連続点はスキャン内に障害物などがあつたり、ガラス張りの環境で多く発生する。キーポイント周辺の点にこれが多くあると、位置合わせに有用な特徴量が得にくくなるため、位置合わせの難易度が増す。

算出した $raw C^t$ に対して主成分分析を行い、次元圧縮した $PCA C^t$ を選択的統合処理で用いる地形複雑度とする。このとき「 $PCA C^t$ の次元数」は累積寄与率が 90% を超える最小の次元数とする。

3.2 選択的統合処理

選択的統合処理では、時刻 t までのスキャンとそのキーポイントの集合 V^t と地形複雑度 $PCA C^t$ に基づいて、現在の時刻 t のスキャンに環境の形状特徴が十分に含まれているか否かを

判断して、不十分であれば現在のスキャンと過去のスキャンを統合して局所地図を生成し、さらに、生成された局所地図に形状特徴が十分含まれているか否かを判断して、生成される局所地図に形状特徴が十分に含まれていると判定されるまで、判断・統合処理を繰り返し、形状特徴が十分に含まれた局所地図を生成する。このとき、過去のスキャンが十分にない場合は、その時点で生成されている局所地図を出力とする（例えば $t = 1$ では局所地図は生成しない）。Algorithm 1 は、選択的統合処理の詳細な処理手順を表している。

Algorithm 1 SelectiveMerge($V^t, PCA C^t$)

Require: $V^t = \{(P^1, K^1), \dots, (P^t, K^t)\}$ /*時刻 t までのスキャンとそのキーポイントの集合*/
 $PCA C^t$ /*時刻 t のスキャンの地形複雑度*/
Ensure: cP^t, cK^t /*形状特徴の十分なスキャンとそのキーポイントの集合*/
1: $cP^t, cK^t \leftarrow \text{popScan}(V^t)$
2: **for** $s = t - 1$ **to** 1 **do**
3: /*スキャン形状の判断*/
4: **if** ShapeJudge($PCA C^t$) = Yes **then**
5: **return** cP^t, cK^t
6: **else**
7: $P^s, K^s \leftarrow \text{popScan}(V^s)$
8: $local q \leftarrow \text{CIF-basedScanMatching}(cP^t, cK^t, P^s, K^s)$
9: $P^s \leftarrow \text{Transform}(P^s, local q)$
10: $merge P^t \leftarrow \text{OrderPreservingMerge}(cP^t, P^s)$
11: /*前処理*/
12: $cP^t, cB^t, cK^t \leftarrow \text{Preprocessing}(merge P^t)$
13: /*地形複雑度の計算*/
14: $PCA C^t \leftarrow \text{CalculateTerrainComplexity}(cP^t, cK^t, cB^t)$
15: **end if**
16: **end for**
17: **return** cP^t, cK^t

以下では、この Algorithm 1 内の関数について説明する。

関数 popScan() は、LILO 型データ構造から、pop して最後の要素の一つ取り出す関数である。したがって、同じ LILO 型のデータ構造がこの関数入力となっている場合、pop されるごとに一時刻前のデータが取り出されることになる。この関数によって、 V^t から取り出されたスキャンとキーポイントの集合の組は、 cP^t, cK^t に保存しておく。また、この関数の入力が、時刻 s までのスキャンとそのキーポイントの集合 V^s (ただし、 t は現在時刻を表し、時刻 s には、 $t - 1$ から順に過去にさかのぼって代入される) の場合、取り出されるデータを、過去のスキャンとキーポイントの集合の組 P^s, K^s に保存しておく。

関数 ShapeJudge() は、地形複雑度 $PCA C^t$ を入力として、 $PCA C^t$ のもとのスキャン cP^t に形状特徴が十分含まれている (Yes) か、含まれていない (No) かを判断する関数である。この判断を下すために、本研究では、SVM に基づく識別器を用いる^{††}。関数 ShapeJudge() の識別器の実体は、学習された SVM に基づく識別器である。この SVM に基づく識別器の学習は、次のようにして行う。まず、通常の CIF-based スキャンマッチング法 [5] を用いて、大域的自己位置推定する環境でスキャンマッチングに成功するスキャン、失敗するスキャンを多数収集し、それぞれのスキャンに対して、スキャンマッチングに成功する・失敗するラベルを付与して、先述した地形複雑度を求める。こうし

[†]本論文では、実験を繰り返した結果、 $m = 9$ と決めた。

^{††}SVM は、教師あり学習を行う学習器の一つで、2 クラスの分類を行う。SVM では、複雑な分布をしている学習データであっても、カーネル関数を用いて高次元の空間に写像することで、線形モデルとして扱うことができる。これを用いることで、地形複雑度が複雑な分布をしていても 2 クラスの識別が可能になる。

て、スキャンマッチングに成功する・失敗するラベルと、それらに対応した地形複雑度の組み合わせのデータを作成する。その後、このデータにおいて、スキャンマッチングに成功するラベルを形状特徴が十分であるラベルに替え、スキャンマッチングに失敗するラベルを形状特徴が不十分であるラベルに替えて、これを SVM に基づく識別器の訓練データとして用いて、識別器を学習する。関数 ShapeJudge() の識別器により、スキャン ${}^cP^t$ の形状特徴が十分と判断された場合は、関数 SelectiveMerge() は、選択的統合処理の結果として、スキャン ${}^cP^t$ とキーポイントの集合 ${}^cK^t$ の組を出力する。関数 ShapeJudge() の識別器により、スキャン ${}^cP^t$ の形状特徴が不十分と判断された場合は、次のような処理を順次実行していく。まず、時刻 s までのスキャンとそのキーポイントの集合 V^s (ただし、 t は現在時刻を表し、時刻 s は、 $s < t$ である。) から、過去のスキャンとキーポイントの集合の組 P^s, K^s を取り出す。その後、関数 CIF-basedScanMatching() を用いて、現在のスキャン ${}^cP^t$ と過去のスキャン P^s の間の相対姿勢 ${}^{local}q = (\Delta\tilde{x}, \Delta\tilde{y}, \Delta\tilde{\theta})$ を求める。関数 CIF-basedScanMatching() は我々が提案している CIF-based スキャンマッチング法で、詳細は 3.3 節で述べる。

Algorithm 2 OrderPreservingMerge(${}^cP^t, P^s$)

Require: ${}^cP^t = \{{}^c p_1^t, \dots, {}^c p_{N^t}^t\}$, $P^s = \{p_1^s, \dots, p_{N^s}^s\}$
Ensure: ${}^{merge}P^t$

- 1: $L \leftarrow \text{RangeSearch}({}^cP^t, P^s)$
- 2: **if** ($L(1) \neq \emptyset$ **and** $L(1) > 1$) **then**
- 3: /* ${}^c p_1^t$ 以前にある P^s 内の点を追加 */
- 4: **for** $j = 1$ to $L(1) - 1$ **do**
- 5: ${}^{merge}P^t \leftarrow \text{addPoint}(p_j^s)$
- 6: **end for**
- 7: **end if**
- 8: ${}^{merge}P^t \leftarrow \text{addPoint}({}^c p_1^t)$
- 9: **for** $i = 1$ to ${}^cN^t - 1$ **do**
- 10: **if** ($L(i) \neq \emptyset$ **and** $L(i+1) \neq \emptyset$ **and** $(L(i+1) - L(i)) > 1$) **then**
- 11: /* ${}^c p_i^t$ と ${}^c p_{i+1}^t$ の間にある P^s 内の点を追加 */
- 12: **for** $j = L(i) + 1$ to $L(i+1) - 1$ **do**
- 13: ${}^{merge}P^t \leftarrow \text{addPoint}(p_j^s)$
- 14: **end for**
- 15: **end if**
- 16: /* 上記以外の場合は、 ${}^c p_{i+1}^t$ を追加 */
- 17: ${}^{merge}P^t \leftarrow \text{addPoint}({}^c p_{i+1}^t)$
- 18: **end for**
- 19: **if** ($L({}^cN^t) \neq \emptyset$ **and** $L({}^cN^t) < N^s$) **then**
- 20: /* ${}^c p_{N^t}^t$ 以降にある P^s 内の点を追加 */
- 21: **for** $j = L({}^cN^t) + 1$ to N^s **do**
- 22: ${}^{merge}P^t \leftarrow \text{addPoint}(p_j^s)$
- 23: **end for**
- 24: **end if**

関数 Transform() は、関数 CIF-basedScanMatching() により求められた相対位置姿勢 ${}^{local}q$ を用いて、スキャン P^s のデータをスキャン ${}^cP^t$ のデータの座標系におけるスキャン P^s に変換する。

関数 OrderPreservingMerge() は、 ${}^cP^t$ と P^s を入力して、これら二つのスキャンを点列の順序を保存して統合したスキャン ${}^{merge}P^t$ を出力する。この統合したスキャン ${}^{merge}P^t$ が局所地

図になる。Algorithm 2 は、関数 OrderPreservingMerge() の詳しい処理手順を示している。また、Fig. 3 は、二つのスキャンを統合したスキャン (局所地図) が生成される過程を表している。

関数 OrderPreservingMerge() の中で、 L は、 ${}^cP^t$ 内の点と同一の位置に存在すると見なせる P^s 内の点の番号を保存するためのリストである。初期状態で、 ${}^{merge}P^t = \{\emptyset\}$ かつ $L = \{\emptyset, \dots, \emptyset\}$ である。

関数 RangeSearch() は、 ${}^cP^t$ 内の点 ${}^c p_i^t$ を中心として距離 ϵ 内 (Fig. 3 内で丸で囲んだ部分) に存在する P^s 内の点の番号を、 ${}^cP^t$ 内のすべての点について求めて、その番号をリスト L に保存する。距離 ϵ 内に存在する点がなければ、その点の番号は $\emptyset$ となる。

まず、 ${}^cP^t$ 以前にある P^s 内の点を ${}^{merge}P^t$ に追加する。次に ${}^cP^t$ を ${}^{merge}P^t$ に追加していく。このとき、 ${}^cP^t$, ${}^cP^{t+1}$ 間に P^s 内の点があれば順に ${}^{merge}P^t$ に追加する。その後 ${}^cP^t$ 以前にある P^s 内の点を ${}^{merge}P^t$ に追加し ${}^{merge}P^t$ を得る。

そして最後に、スキャン ${}^{merge}P^t$ を新たな入力として、前処理、地形複雑度の算出、スキャンの形状判断を行う。

最終的に、このような統合処理をスキャンの形状特徴が十分になるまで繰り返し、形状特徴が十分なスキャン (局所地図) とそのスキャンのキーポイントの集合の組 ${}^cP^t, {}^cK^t$ を生成する。

3.3 CIF-based スキャンマッチング法

Algorithm 3 は、これまで我々が提案してきた CIF-based ス

Fig. 3 How to perform order-preserving merge

キャンマッチング法（関数 `CIF-basedScanMatching()`）の詳細な処理手順を表している．関数 `CIF-basedScanMatching()` は、二つのスキャン ^{ref}P , ^{cur}P と各スキャンにおけるキーポイントの集合 ^{ref}K , ^{cur}K を入力として与えると、 ^{ref}P に対する ^{cur}P の相対的な位置姿勢 $q = (\Delta\tilde{x}, \Delta\tilde{y}, \Delta\tilde{\theta})$ を求める．このとき、相対的な位置姿勢の初期値を与える必要がない．

Algorithm 3 `CIF-basedScanMatching( $^{ref}P$ ,  $^{ref}K$ ,  $^{cur}P$ ,  $^{cur}K$ )`

Require: ^{ref}P , ^{ref}K , ^{cur}P , ^{cur}K

Ensure: $q = (\Delta\tilde{x}, \Delta\tilde{y}, \Delta\tilde{\theta})$

```

1: /*Step1: 特徴量の算出*/
2:  $^{ref}H \leftarrow \text{CalculateCIFdescriptors}(^{ref}P, ^{ref}K)$ 
3:  $^{cur}H \leftarrow \text{CalculateCIFdescriptors}(^{cur}P, ^{cur}K)$ 
4: /*Step2: 対応点探索*/
5:  $T \leftarrow \text{FindCorrespondingPoints}(^{ref}K, ^{cur}K, ^{ref}H, ^{cur}H)$ 
6: /*Step3: 粗な位置姿勢推定*/
7:  $^{rough}q \leftarrow \text{RoughMatching}(T)$ 
8: /*Step4: 詳細マッチング*/
9:  $q \leftarrow \text{ICP}(^{rough}q, ^{ref}P, ^{cur}P, T)$ 
10: return  $q$ 

```

関数 `CIF-basedScanMatching()` によって大域的自己位置推定を実行する際には、 ^{ref}P として ^{Map}P （大域的地図に相当する）、 ^{cur}P として $^{c}P^t$ を用いる．選択的統合処理の中で関数 `CIF-basedScanMatching()` を使用する際には、 ^{ref}P として $^{c}P^t$ （局所地図に相当する）、 ^{cur}P として $^{P^s}$ を用いる．

CIF-based スキャンマッチング法は四つのステップ ① 特徴量の算出、② 対応点探索、③ 粗な位置姿勢推定、④ 詳細マッチング、からなる．

ステップ ① では、入力された各スキャン ^{ref}P , ^{cur}P の CIF 特徴量を算出する．ステップ ① に相当する関数 `CalculateCIFdescriptors()` は、スキャンとキーポイントの集合を入力として与えると、各キーポイントにおける CIF 特徴量 outputs. Algorithm 3 中の変数 ^{ref}H , ^{cur}H は各スキャン ^{ref}P , ^{cur}P の CIF 特徴量の集合を表している．

ステップ ② では、ステップ ① で求めたキーポイントにおける CIF 特徴量とキーポイント間の幾何学的な拘束を利用してスキャン ^{ref}P のキーポイントに対応するスキャン ^{cur}P のキーポイントを求める．

ステップ ② に相当する関数 `FindCorrespondingPoints()` は、二つのスキャンにおけるキーポイントの集合と、キーポイントにおける CIF 特徴量の集合を入力として与えると、それぞれのスキャンにおける 3 個のキーポイントの組み合わせをすべて求め、二つのスキャンにおいて CIF 特徴量が最も一致するキーポイントの組み合わせを、二つの対応するキーポイントの組み合わせ $T = \{(^{ref}P_{\alpha 1}, ^{ref}P_{\alpha 2}, ^{ref}P_{\alpha 3}), (^{cur}P_{\beta 1}, ^{cur}P_{\beta 2}, ^{cur}P_{\beta 3})\}$ を出力する．ここで、 αi と $\beta i (i = 1 \sim 3)$ が対応している二つのキーポイントの番号を表している．

ステップ ③ では、ステップ ② で求めた二つの対応するキーポイントの組み合わせを用いて、文献 [21] [22] に基づいて、 ^{ref}P に対する ^{cur}P の相対的な位置姿勢を求める．

ステップ ③ に相当する関数 `RoughMatching()` は、二つの対応するキーポイントの組み合わせ T を入力として与えると、

^{ref}P に対する ^{cur}P の相対的な位置姿勢 $^{rough}q$ を出力する．なお、ここで求められる位置姿勢 $^{rough}q$ は、二つの対応するキーポイントの組み合わせのみのデータしか用いていないため、二つのスキャンを構成するスキャン点どうしの多くの部分を位置合わせできるような解となっていないので、相対的な位置姿勢の近似解となる．

ステップ ④ では、ステップ ③ で求めた近似解 $^{rough}q$, スキャン ^{ref}P , ^{cur}P , 二つの対応するキーポイントの組み合わせ T を用いて、ICP アルゴリズム [23] [24] によって、 ^{ref}P に対する ^{cur}P の相対的な位置姿勢の詳細解 q を求める．ステップ ④ に相当する関数 `ICP()` は、ICP アルゴリズム [23] [24] を実行する関数である．こうして求められた位置姿勢の詳細解 q が、関数 `CIF-basedScanMatching()` の出力となる．

4. 検証実験

SICK 社のレーザレンジファインダ（LMS100: 視野 270 [deg], 角度分解能 0.25 [deg], 測定可能距離 20 [m]）を用いて和歌山大学システム工学部 A 棟 1 階のフロア（65 [m] × 45 [m]）でスキャンを取得し、スキャン 66 個を用いて総スキャン点数 13,003 個の地図を生成して実験に使用した．Fig. 4 は、地図を構成するすべてのスキャン点を可視化したものである．

4.1 地形複雑度によるスキャンの形状特徴の判断の検証

環境地図内で、地図作成に用いたスキャンとは別に 1,482 地点のスキャンを収集し、そのスキャンと、地形複雑度を用いて CIF-based スキャンマッチング法による自己位置推定を行った際の結果（CIF-based スキャンマッチングに成功する・失敗するというラベル）を記録して、SVM に基づく識別器のための訓練データを作成した．訓練データ作成時のラベルの割り振り は自己位置推定の結果を人が見て判断する．

SVM に基づく識別器を用いて、環境の形状特徴の有無を判断する実験を行った．SVM に基づく識別器に入力する地形複雑度は、主成分分析を用いて五次元に次元圧縮してから入力した．このときの累積寄与率は 92.4% である．実験では、SVM で用いるカーネル関数として、RBF カーネルを用い、カーネルパラメータはグリッドサーチによって決定した．収集された 1,482 個のデータ（地形複雑度とラベル（形状特徴の十分・不十分）の組）はラベルの数に偏りがあったため（形状特徴十分: 948 個、形状特徴不十分: 534 個）、ランダムに形状特徴十分・不十分の

Fig. 4 The map of 1st floor in building of Wakayama University

データを 500 個ずつ選び SVM に基づく識別器の訓練に使用した。10 分割交差検証によって、訓練した識別器の識別率を評価したところ、識別率は 83% であった。スキャン内の環境の形状特徴の有無の判断を誤る場合に二つの種類がある（(1) 特徴が十分な場合でも不十分と判断する誤り、(2) 特徴が不足するにもかかわらず特徴が十分と判断する誤り）。(1) が起こる状況では、形状特徴が十分あるので自己位置推定は可能であると考えられる。(2) の場合は自己位置推定に失敗してしまうことが予想される。今回の実験では (1) が起こる確率は全体の 9% で、(2) が起こる確率は全体の 8% であった。形状特徴が十分にある場合自己位置推定に成功するとみなせる。このことから、識別の成功した 83% では自己位置推定に成功し局所地図生成に用いるスキャン数が最小となる。(1) が起こる 9% では自己位置推定に成功するが局所地図生成に用いるスキャン数が最小にならない。(2) の起こる 8% では自己位置推定が失敗する。したがって、環境の形状特徴の有無を判断する識別率が 83% であったとしても、そのような識別器を用いて自己位置推定に成功する確率は、約 $83 + 9 = 92\%$ となる。この結果から、スキャンから地形複雑度を算出し、SVM 識別器を用いてスキャン内の環境の形状特徴の十分・不十分の判断する方法は、自己位置推定法にとって有効であると考えられる。

識別器の訓練に必要な訓練データは、本研究では、人間が判断してラベル（環境の形状特徴が十分か不十分か）の付与を行った。しかし、環境により訓練データが膨大になり、ラベル付与に対するコストが増す可能性がある。ラベルの付与を自動化させることは今後の課題であるが、次のような手順で実現できると考えられる。① 地図内でランダムに多数の場所を選び、その地点において観測されるスキャンを生成する（地図データ内から 1 時刻分のスキャンデータを選択する）。② 生成したスキャンを入力データとして CIF-based スキャンマッチング法による自己位置推定を行う。③ はじめに選択した場所（位置姿勢）と推定された位置姿勢が一致しているか否かを判定して、ラベルを付与する。

4.2 選択的統合処理の検証

Fig. 4 に示した環境でロボットを走行させ、その環境内の 120 地点で提案手法による大域的自己位置推定を行った。その際に、各地点で選択的統合処理によって作成された局所地図を構成するスキャンの個数を調査した。Fig. 5 は、その調査結果を、縦軸を地点数、横軸をスキャンの個数としたヒストグラムとしてまとめたものを表している。このヒストグラムに表されているように、局所地図を作成せずにその地点におけるスキャンのみで大域的自己位置推定が行えたのは 79 地点で、その他の地点では複数のスキャンを統合して局所地図を作成していることが分かる。複数のスキャンの個数としては 2~28 個まで分布しており、地点に応じて局所地図作成に用いるスキャンの個数が変化していることが分かる。

4.3 提案手法による大域的自己位置推定の検証

提案手法による大域的自己位置推定の結果の例として、Fig. 4 に示した場所 A と場所 B における結果を示す。場所 A、場所 B において 1 時刻に観測されるスキャンは、それぞれ、Fig. 6 (a) と Fig. 7 (a) に示されている。Fig. 6 (a) に示されているスキャ

Fig. 5 Histogram of the number of scans used for generating a local map

Fig. 6 The example of a raw and a merged scan at location A

Fig. 7 The example of a raw and a merged scan at location B

ンは、壁面に大きな凹凸のある直線的な廊下（場所 A）で取得されたもので、1 時刻のスキャンデータの中にオクルージョンがあり、形状特徴も少ない。Fig. 7 (a) に示されているスキャンは、ガラス張りの大きな広間（場所 B）において取得されたもので、1 時刻のスキャンデータ中にレーザが透過する箇所が複数存在するため、不連続点を多く含み、形状特徴を得るのが難しい。これらの例で示されているような、1 時刻のスキャンのみでは大域的自己位置推定が困難な場所について、提案手法が有効であるか検証した。

場所 A において提案手法により大域的自己位置推定を行った際には、3 時刻分のスキャンを統合してできた局所地図が生成された。Fig. 6 (b) は生成された局所地図を表している。また、場所 B において提案手法により大域的自己位置推定を行った際には、8 時刻分のスキャンを統合してできた局所地図が生成された。Fig. 7 (b) は生成された局所地図を表している。それぞれの

Fig. 8 A result of CIF-based scan matching without merging at location A

Fig. 9 A result of CIF-based scan matching without merging at location B

Fig. 10 A result of our method at location A

場所において、1時刻のみのスキャン（各図（a））では破線で囲んだ部分にスキャン点が存在しない部分（オクルージョン）がある。一方、各図（b）の生成された局所地図の破線で囲んだ部分には、スキャン点が存在していることが分かる。

Fig. 4 に示した場所 A と場所 B において、1時刻のみのスキャンを用いて CIF-based スキャンマッチング法により大域的自己位置推定を行い、推定された自己位置に従って環境地図にスキャンを重ね合わせた結果を Fig. 8, Fig. 9 に示す。これらの図内の小さな点は環境地図を表し、図内の破線で囲まれた部分には、1時刻分のスキャンが大きい点の集合として表されている。

場所 A, 場所 B においては、1時刻分のスキャンだけでは大域的自己位置推定に失敗している。

Fig. 10, Fig. 11 は、それぞれ、Fig. 4 に示した場所 A と場所 B において、提案手法により Fig. 6 (b), Fig. 7 (b) の局所地図を生成した後、これらの局所地図と Fig. 4 に示した環境地図

Fig. 11 A result of our method at location B

Fig. 12 A result of MCL method at location A

Fig. 13 A result of MCL method at location B

との位置合わせを行った結果を表している。これらの図内の小さな点は環境地図を表し、図内の破線で囲まれた部分に、生成された局所地図が大きい点の集合として表されている。

この結果を見ると、二つの場所で、それぞれ、3時刻分と8時刻分のスキャンから生成した局所地図と環境地図とで正しく位置合わせが行えており、提案手法によって正しく大域的自己位置推定が行えていることが分かる。

Fig. 12, Fig. 13 は、それぞれ、Fig. 4 内に示した場所 A, 場所 B において、ロボットが自己位置を見失った（ロボット誘拐問題が発生した）と想定して、複数時刻のスキャンを入力して MCL 法によって大域的自己位置推定を行った結果を表している。初期状態で、環境地図内に 5,000 個のパーティクルをランダムに散布し、スキャンを順次入力後、散布されたパーティクルが1箇所に収束するまでに入力したスキャンの数を調査し

Table 1 Difference between actual value and estimated value

	$\Delta\tilde{x}[m]$	$\Delta\tilde{y}[m]$	$\Delta\tilde{\theta}[deg]$
location A	+0.053	+0.02	+0.5
location B	-0.08	-0.29	+0.4

Fig. 14 The map of 7th floor in building of Wakayama university

た. Fig. 12 の破線で囲んだ部分に示すように, 場所 A では, 大域的自己位置推定に失敗し, パーティクルが 1 箇所に収束するまでに入力したスキャン数は 4 であった. Fig. 13 の破線で囲んだ部分に示すように, 場所 B では大域的自己位置推定に成功し, パーティクルが収束するのに入力したスキャン数は, 14 であった.

これらの結果を見れば分かるように, この二つの場所では, 提案手法を用いた場合のほうが MCL 法よりも自己位置推定に要するスキャン数が少なく済むことが分かる. また, MCL 法で大域的自己位置推定に失敗するような場合でも, 提案手法では自己位置推定に成功することが分かる.

提案手法によって位置合わせを行った結果得られた位置姿勢と真値[†]との比較を行った. **Table 1** は, Fig. 10, Fig. 11 に示した位置合わせの結果から得られた位置姿勢と真値との誤差を表している.

この結果から分かるように, 提案手法で推定した位置姿勢は誤差が小さいことが分かる.

4.4 未学習な環境における提案手法の有効性の検証

ある環境のデータで訓練した識別器が他の環境のデータに対して適用可能か検証するために, 和歌山大学システム工学部 A 棟 7 階のフロアでスキャンを取得し, スキャン 150 個を用いて総スキャン点数 25,018 個の地図を生成して実験に使用した. **Fig. 14** は, 地図を構成するすべてのスキャン点を可視化したものである.

Fig. 14 内の場所 C と場所 D において, 提案手法により大域自己位置推定を行う. **Fig. 15** (a), **Fig. 15** (b) は, それぞれ, 場所 C (廊下の曲がり角), 場所 D (フロア内の一室の入り口) において取得されたスキャンを表している. いずれのスキャンも, 識別器を訓練するとき用いたデータには存在しなかったスキャンである. 場所 C, 場所 D において, 提案手法によって大域的自己位置推定した結果を **Fig. 16** に示す. この図内の破

(a) A scan at location C (b) A scan at location D

Fig. 15 Two examples of raw scans

(a) A result of our method at location C (b) A result of our method at location D

Fig. 16 Two results of our method at location C and D

線で囲まれた部分に, 生成された局所地図が示されている.

Fig. 16 (a) に示すように, 場所 C では, 4 時刻分のスキャンから局所地図を作り, 自己位置推定が行われ, 正しい結果が得られた. 一方, Fig. 16 (b) に示すように, 場所 D では, 自己位置推定に用いられたスキャン数は一つであった. このスキャンに対しては形状特徴が不十分なものにもかかわらず, 識別器には形状特徴が十分と判断された. この結果から, 識別器訓練時には存在しなかった形状のスキャンに対しては統合の必要・不要の判断を誤る可能性があり, 最終的に大域的自己位置推定を誤ってしまうことが分かる.

5. まとめと今後の課題

本研究では, スキャンの地形の特徴を表す“地形複雑度”を提案し, それに基づいてそのスキャンに環境の形状特徴が十分に含まれているか否かを判断して, 用いるスキャンの数を変化させて局所地図を生成する方法“選択的統合処理”を提案した. また, 生成した局所地図を使用して大域のスキャンマッチングを行う方法を提案した. 実環境の地図を用いて大域的自己位置を行う実験結果から, 地形複雑度に基づいてスキャンに含まれる形状特徴が十分か否かを RBF カーネルを用いた SVM に基づく識別器により判断できること, 選択的統合処理により位置合わせに用いるスキャンの数を限定せずに局所地図を生成できること, また, その局所地図を用いて大域的自己位置を頑健に行えることを示した.

比較実験により, 提案手法を用いることで, MCL 法を用いるよりも大域的自己位置推定に用いるスキャン数が少なくなること示した. また, レーザが透過して特徴を得ることが難しいガラス張りの環境や, 得られる特徴が少ない廊下のような環境においても, 提案手法を用いることで, 正しく大域的自己位置推定を行えることを確認した. 提案手法は, 1 時刻のスキャン

[†] 地図とスキャンの位置合わせを手動で行い, その位置姿勢を真値とした.

内では位置合わせに有効な特徴が少ないが、過去にさかのぼった複数の時刻におけるスキャン内に位置合わせに有効となる特徴が存在する環境では、有効である。しかし、過去のスキャンをいくらさかのぼっても特徴が存在しないような環境（例えばまっすぐな廊下が延々と続く環境）では、提案手法による大域的自己位置推定は失敗することが予想されるが、これに対応することは今後の課題である。また、識別器を訓練するときに用いたデータ中に存在しない形状が現れた場合、識別器が判断を誤る可能性があるため、識別器は、多様な形状を含む環境で取得したデータを用いて訓練する必要がある。スキャンに含まれる形状特徴が十分か否かを判断する際の識別率をさらに高めるために、新たな地形複雑度について検討する予定である。また、提案手法を、二次元レーザスキャンを用いた大域的自己位置推定に用いるだけでなく、三次元レーザスキャンを用いた大域的自己位置推定の問題に適用することを検討する予定である。

謝 辞 本研究の一部は科学研究費（基盤研究（C）No. 23500243）の助成を受けたものである。

参 考 文 献

- [1] J.S. Gutmann, T. Weigel and B. Nebel: "Fast, Accurate, and Robust Self-Localization in Polygonal Environments," Proc. of IROS '99, pp.1412-1419, 1999.
- [2] P. Jensfelt and S. Kristensen: "Active Global Localization for a Mobile Robot Using Multiple Hypothesis Tracking," IEEE Trans. on Robotics and Automation, vol.17, no.5, pp.748-760, 2001.
- [3] 中村恭之, 脇田翔平: "2D スキャンデータの合同変換に不変な特徴量 (CIF) を用いたロバストなスキャンマッチング", 第 19 回ロボティクスシンポジウム, 6C3, pp.592-598, 2014.
- [4] 中村恭之, 脇田翔平: "合同変換に不変な特徴量 (CIF) とキーポイント間の幾何学的拘束に基づいたロバストなスキャンマッチング法の提案", 計測自動制御学会論文集, vol.51, no.5, pp.309-318, 2015.
- [5] 脇田翔平, 中村恭之: "スキャンデータの順序保存統合を用いた CIF ベーススキャンマッチング法", ロボティクス・メカトロニクス講演会 '15, 2A2-M09, 2015.
- [6] F. Dellaert, D. Fox, W. Burgard and S. Thrun: "Monte Carlo Localization for Mobile Robots," Proc. ICRA '99, pp.1322-1328, 1999.
- [7] F. Campos, L. Correia and J. Calado: "Global Localization with Non-quantized Local Image Features," Journal of Robotics and Autonomous Systems, vol.60, no.8, pp.1011-1020, 2012.
- [8] 阪東茂, 原祥亮, 坪内孝司: "2 次元測域センサデータの空間周波数解析に基づく大域的自己位置推定", 第 18 回ロボティクスシンポジウム, pp.163-168, 2013.
- [9] K. Konolige and K. Chou: "Markov localization using correlation," Proc. IJCAI '99, pp.1154-1159, 1999.
- [10] E. Olson: "Real-time correlative scan matching," Proc. of ICRA '09, pp.4387-4393, 2009.
- [11] J. Gutmann and D. Fox: "An Experimental Comparison of Localization Methods Continued," Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS '02), pp.454-459, 2002.
- [12] 上田隆一, 新井民夫, 浅沼和範, 梅田和昇, 大隅久: "パーティクルフィルタを利用した自己位置推定に生じる致命的な推定誤りからの回復法", 日本ロボット学会誌, vol.23, no.4, pp.466-473, 2005.
- [13] A. Diosi and L. Kleeman: "Laser scan matching in polar coordinates with application to SLAM," Proc. of IROS '05, pp.3317-3322, 2005.
- [14] 友納正裕: "ユークリッド変換に不変な特徴量を用いた二次元大域スキャンマッチング方式", 日本ロボット学会誌, vol.25, no.3, pp.390-401, 2007.
- [15] 藤井祐介, 大矢晃久, 坪内孝司: "人混み中での移動ロボットの自己位置推定に有効な測域センサデータ処理法の提案—蓄積処理と近傍点削除による壁面の抽出—", ロボティクス・メカトロニクス講演会 '14, 2A2-Q01, 2014.
- [16] 原祥亮, 坪内孝司, 大島章: "確率的に蓄積したスキャン形状により過去を考慮した Rao-Blackwellized Particle Filter SLAM", 日本機械学会論文集, vol.82, no.834, p.15-00421, 2016.
- [17] 田下裕一, 中村恭之: "ロバストなスキャンマッチングのための合同変換に不変な特徴量 (CIF) の提案", ロボティクス・メカトロニクス講演会 '13, 1A2-H10, 2013.
- [18] V.N. Vapnik: Statistical Learning Theory. J. Wiley, 1998.
- [19] V.N. Vapnik: The Nature of Statistical Learning Theory (Second Edition). Springer, 1999.
- [20] G.A. Borges and M.J. Aldon: "Line extraction in 2D range images for mobile robotics," Journal of Intelligent and Robotic Systems, vol.40, pp.267-297, 2004.
- [21] K. Lingemann, H. Surmann, A. Nuchter and J. Hertzberg: "Indoor and outdoor localization for fast mobile robots," Proc. of IROS '04, pp.2185-2190, 2004.
- [22] 中村恭之: "制約条件付最小 2 乗法に基づく高速な局所型スキャンマッチング法", 日本ロボット学会誌, vol.28, no.5, pp.648-657, 2010.
- [23] P.J. Besl and N.D. 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.
- [24] F. Lu and E. Milios: "Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans," Journal of Intelligent and Robotic Systems, vol.18, pp.249-275, 1997.

脇田翔平 (Shohei Wakita)

2015 年和歌山大学システム工学研究科博士前期課程修了。2016 年和歌山大学システム工学研究科博士後期課程入学, 現在に至る。この間, 二次元レーザスキャナを用いた移動ロボットの自己位置推定法の研究に従事。(日本ロボット学会学生会員)

中村恭之 (Takayuki Nakamura)

1996 年大阪大学大学院工学研究科博士後期課程修了。同年, 日本学術振興会特別研究員。1997 年奈良先端科学技術大学院大学情報科学研究科助手, 2002 年和歌山大学システム工学部情報通信システム学科助教授, 2013 年同大学同学部同学科教授となり現在に至る。この間, 1996 年から約 1 年間米国ブラウン大学計算機科学科客員研究員, 2007 年から約 1 年間米国カーネギーメロン大学ロボティクス研究所客員研究員。ロボットビジョン, 視覚情報に基づくロボット学習システム, 移動ロボットの自己位置推定法の研究に従事。博士 (工学)。計測自動制御学会会員。

(日本ロボット学会正会員)