2Q-05

大規模点群データ分析のためのデータベースの検討

笠井 雄太 † 石川 佳治 †† 杉浦 健人 ††
† 名古屋大学工学部電気・電子情報工学科 †† 名古屋大学大学院情報学研究科

1 はじめに

近年，自動運転や移動ロボットにおける周辺の状況のセンシングや，飛行機からの計測をもとに構築される3次元の地形図構築などにおいて，**点群データ**の取得と活用が活発に行われている．点群データは**ポイントクラウド**（point cloud）とも呼ばれ，専用のライブラリやソフトウェアの構築が行われている [6]．また，ロボットシステムの開発のためのフレームワークである ROS にも点群データのモジュールが含まれている [5]．このように点群データの利用が拡大するに伴い，大量の点群データをどのように管理し，点群データの利活用を図るかという問題がでてきた．

このような背景を受け，本研究ではデータベース管理システム（DBMS）を用いた点群データを管理方式について開発し，その実装を行う．具体的には，PostgreSQL [8] および点群データに関するライブラリ等を用いて，点群データの蓄積のための実装を行う．加えて，具体的な応用として，DBMS への点群データ格納時に前処理を行うことによる，問合せ時の点群マッチングの高速化のアプローチについて述べる．

2 関連研究

簡単に関連研究について述べる．点群データを DBMS で管理するというアプローチについては，まだ研究の萌芽的な状況にある．積極的な取組みを行っているのはインメモリ DBMS MonetDB のプロジェクトであり，点群データの管理に関する手法の比較 [4] や，インメモリの点群データの処理方式の開発 [3] を行っている．点群データに適したデータ構造などの開発や，ベンチマークの開発 [10] も行っており，一つの研究の方向性を示している．

DBMS を用いた点群データの管理に関する簡単なアイデアの提示については，[1, 9] においても行われているが，具体的な技術の詳細や実装の内容については不明である．

本研究では，点群データに関する既存のライブラリおよびソフトウェアの活用を行うことを一つの目的とする．また，対象とする点群データライブラリと DBMS との連携を密にして，問合せ処理時などにおけるシステムの効率化を図る．

Construction of a Database System for Analyzing Large Point Cloud Data
Yuta Kasai†, Yoshiharu Ishikawa††, Kento Sugiura††
† Department of Information Engineering, School of Engineering, Nagoya University
†† Graduate School of Informatics, Nagoya University

3 システムの構成

ここではシステムの構成について述べる．システムの構成を図1に示す．ただし，本稿執筆中の時点における予定の部分も含まれるため，最終的な実装は異なる可能性がある．

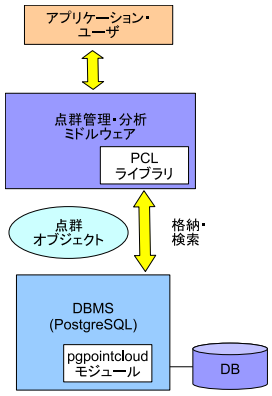

図 1: システムの構成

3.1 pgpointcloud モジュールの活用

PostgreSQL において点群データを蓄積するため，点群データのための PostgreSQL 拡張モジュールである pgpointcloud [7] を用いる．このモジュールは `PcPoint` と呼ばれる点群のためのデータ型を提供する．`PcPoint` データ型は複数の次元を取ることができ，また，各次元に対して複数の属性を持つことができる．実世界における点群データにはさまざまな用途があり，単なる座標値だけでは十分に情報を表現できないためである．要求に応じて `PcPoint` 型のフォーマットが柔軟に対応できる必要があるため，データ型の構造をスキーマにより記述する仕組みが設けられている．

一つ一つの点である `PcPoint` 型のデータをそのまま個別に用いるのは面倒な場合もある．pgpointcloud では，`PcPoint` 型のデータを集めたオブジェクトを表すための `PcPatch` というデータ型が支援されている．通常，一つの `PcPatch` には，近くに位置する点の集合がまとめられる．

pgpointcloud では，さまざまな関数が定義されており，PostgreSQL の SQL 文の中で使用することができる．具体的には，たとえば以下のような関数が存在する．

- `PcPoint` オブジェクトの値にアクセスするための関数

- PcPoint オブジェクトの集合から PcPatch オブジェクトを作成する関数
- 二つの PcPatch オブジェクトの和集合をとる関数
- 二つの PcPatch オブジェクトの境界が交わるかどうかを判定する関数

データベースの問合せにおいては，PcPatch オブジェクトの ID を用いたオブジェクトの取得，属性に対する条件による検索，オブジェクトの境界が交わるかどうかによる問合せ等が行える．これにより，点群データの格納と基本的な検索が可能となる．

3.2 PCL ライブラリとの連携

今回のシステム開発では，点群データを活用するアプリケーションとの連携を図るため，主要なライブラリの一つである PCL (Point Cloud Library) [6] との密な連携をとる．PCL では，フィルタリング，特徴抽出，特徴点集合の抽出，位置合わせ，セグメンテーションなどの点群データを扱うさまざまな関数が提供されている．DBMS との連携を行うことで，DBMS から検索した点群データを PCL の関数に即座に提供することができる．

次節では，密な連携への対応の試みとして，二つの点群データの位置合わせ処理を含む問合せの高速化に関して述べる．

4 点群データの位置合わせ処理への対応

二つの点群データが与えられたとき，点群がうまく重なり合うようにすることを位置合わせ（registration）と呼ぶ．代表的なアルゴリズムとしては ICP (Iterative Closest Point) アルゴリズムが存在する [2]．正確なマッチングを行うには，計算コストが大きいことが知られている．

本研究では，DBMS にいったん格納した点群データに関しては格納時点で前処理を行い，位置合わせ処理に必要な情報をあらかじめ抽出しておくことにする．これにより，DBMS 内に存在する二つの点群データに対する位置合わせの要求が発生したとき，また，ユーザが与えた点群データとデータベース内の点群データの位置合わせを行う際，その処理が高速化できることになる．

以下ではそのアイデアについて簡単に説明する．PCL における点群データの位置合わせは，以下のステップで実行される．

1. 特徴点集合の抽出 : 点群データから，その点群データの特徴をよく表す点の集合をある種の判断記述を用いて選択する．
2. 特徴記述子の計算 : 特徴点集合をもとに，その点集合の特徴を表現する特徴量を抽出する．
3. 対応関係の推定・棄却 : 特徴のタイプに依存して，特徴のマッチングを行う．また，誤った対応の棄却を行う．
4. 変換の推定 : 位置合わせを点群データの変換処理と捉えて，その推定を行う．

このうち，ステップ 2 までの処理は DBMS に点群データを蓄積する時点において PCL を用いて前もって進めておくことができる．計算した特徴記述子を DBMS に元の点群データと対応づけて格納する．

DBMS 内に格納された点群データに対し位置合わせを行う場合，その点群データに対する特徴記述子を DBMS から取り出し，PCL による位置合わせ処理を行うが，ステップ 2 までの処理はバイパスすることになる．これにより，位置合わせ処理の応答時間の短縮が可能になると考えられる．

5 まとめ

本稿では，大規模点群データを DBMS で管理するためのシステム技術について述べた．今後は実装を含む開発を行っていく予定である．

謝辞

本研究は，JSPS 科研費（16H01722, 18H06461）の助成，および，国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO） の委託業務による．

参考文献

[1] László Dobos, et al. Point cloud databases. In *26th Int'l Conf. on Scientific and Statistical Database Management (SSDBM 2014)*, 2014.

[2] Wikipedia: Iterative closest point. https://en.wikipedia.org/wiki/Iterative_closest_point.

[3] Kostis Kyzirakos, et al. In memory processing of massive point clouds for multi-core systems. In *12th Int'l Workshop on Data Management on New Hardware (DaMon 2016)*, 2016.

[4] Oscar Martinez-Rubi, et al. Benchmarking and improving point cloud data management in MonetDB. *SIGSPATIAL Special*, Vol. 6, No. 2, pp. 11–18, 2014.

[5] 西田健ほか. 実用ロボット開発のための ROS プログラミング. 森北出版, 2018.

[6] PCL: Point Cloud Library. http://www.pointclouds.org/.

[7] Github - pgpointcloud/pointcloud. https://github.com/pgpointcloud/pointcloud.

[8] PostgreSQL: The world's most advanced open source relational database. https://www.postgresql.org/.

[9] Stella Psomadaki. Using a database for dynamic point cloud data management. Master's thesis, Delft University of Technology, 2016. (Graduation Plan).

[10] Peter van Oosterom, et al. Massive point cloud data management: Design, implementation and execution of a point cloud benchmark. *Computer & Graphics*, Vol. 49, pp. 92–125, 2015.