ロードマップを用いたジオソーシャルデータに対する効率的なクローリング手法

井嶋 蒼 [1]　横山 昌平 [2]

Google Places API は，Google Maps に登録されているレストランやホテルにおけるクチコミなどのジオソーシャルデータをクロールすることができる API である．Google Maps は，世界で 10 億人以上のユーザがいるとされ，そのうち約 5000 万人のローカルガイドにより，日々膨大なクチコミが寄せられている．そのため，マーケティングや観光情報の分析において，非常に重要なデータベースになっている．しかし，Google Places API をはじめとした多くのソーシャルデータに対する問い合わせ API は，一日のリクエスト数が制限されており，短期間で大規模なデータを集めることが困難である．そこで，本研究では，一般的に，建物は道路に沿って建てられているという仮説のもと，ロードマップを用いた効率的なクローリング手法を提案する．具体的には，OpenStreetMap から抽出したロードマップを用いて，地物の疎密に応じた効率的なクエリ中心点を決定する．提案手法で集めたデータと網羅的な手法で集めたデータを比較し，考察を行う．

1　はじめに

Google Maps[1] は，Google 社が提供している地図検索サービスである．世界での月間利用者数は 10 億人以上とされており，約 5000 万人以上のローカルガイドが Google Maps の作成を行っている．また，毎日更新されるデータ数は 2500 万個を超える [1]．Google Maps に登録されているデータには，建物に関するクチコミや写真といった詳細な情報が含まれている．そういったデータは多くのローカルガイドによって，日々蓄積されているので，Google Maps は新規性の高い情報を得ることができる非常に重要なデータベースであるといえる．また，地域のお店や観光スポット，ランドマークといった建物に関する新規性の高いデータを分析することはマーケティングや観光といった消費者を主なターゲットとしている分野において必要不可欠である．

個人や他企業といった大小を問わないサードパーティが Google Maps を様々な形で扱えるサービスとして，Google Maps Platform[2] がある．Google Maps Platform は主に Maps API，Routes API，Places API の 3 つの API に分かれている．それらの内の Places API を用いることで Google Maps に登録されている様々な情報をクロールすることができる．Places API を用いてクロールすることができたデータには，単に建物の位置情報などの基本的な情報だけではなく，それに付随するクチコミや写真などといった詳しい情報も含まれる．これらはローカルガイドが作成したジオソーシャルデータである．

一方で，Google Maps のようなジオソーシャルデータを扱うサービスと似ているものとして OpenStreetMap[3] がある．OpenStreetMap は Google Maps などの企業が提供するサービスとは違い，すべて無料で制限なくデータをクロールすることができる．しかし，OpenStreetMap は自由に活用できる地図を作ることを目的としたサービスである．地球上にある建物に関する情報を蓄積するのではなく，地図を作成すること自体に焦点を当てているため，建物に関するクチコミや写真といった詳しい情報を得ることができない．

しかし，OpenStreetMap でクロールすることはできても，Google Maps ではクロールすることができないデータが存在する．それは地図上に描かれているロードマップである．そして，我々は，一般的に，建物は道路に沿って建てられているという点に着目し，ビッグデータをクロールするときにロードマップを活用できるのではないかと考えた．本研究では，無作為にクエリ中心点を決めるのではなく，OpenStreetMap から抽出したロードマップ上にクエリの中心点を決定することで地物の疎密に応じた効率的なクローリング手法を提案する．また，提案手法と比較する手法として範囲をメッシュに区切り，データを網羅的にクロールする手法を採用する．そして，提案手法で集めたデータと網羅的な手法で集めたデータを比較し，考察を行う．

本手法を用いて解決できることは API を用いる際に生じてしまうリクエスト数の制限を回避できることである．マーケティングや観光といった分野でデータ分析を行うとき，たいていは大規模なデータいわゆるビッグデータが必要となる．しかし，Google Places API などをはじめとした企業が提供している多くの問い合わせ API にはリクエスト制限があり，多くのリクエストを短時間で行うと制限がかかってしまい，解除されるまで一定時間待たなければいけない．例えば，Google Places API は一秒あたりのリクエスト数の上限が 100 件である．それを超えてしまうと制限がかかってしまう．そのため，長時間クロールを行っても膨大なデータを得ることが難しい．また，網羅的に無作為にクエリ中心点を決定していると必然的にクエリ回数が多くなり，リクエスト制限にかかってしまう可能性が高くなってしまう．しかし，本手法では，抽出したロードマップに沿って規則的にクエリ中心点を決定するため，クエリ回数を削減することが可能である．そして，クエリ回数を削減することにより，リクエスト制限にかかってしまうリスクを減らし，大規模なデータを効率的にクロールできることに繋がると考える．

また，リクエスト制限の他に API の使用にかかるコストも削減できると考えられる．企業によっては，API を使用するために課金をしなければならない．Google Maps Platform で提供されている Maps API，Routes API，Places API の三つの API は 2018 年 6 月 11 日から料金体系が大きく変更になった．毎月の 200 ド

[1] 非会員　首都大学東京大学院 システムデザイン研究科
ijima-so@ed.tmu.ac.jp
[2] 正会員　首都大学東京大学院 システムデザイン研究科
shohei@tmu.ac.jp

[1] https://www.google.co.jp/maps
[2] https://cloud.google.com/maps-platform
[3] https://www.openstreetmap.org

ル分のリクエストは無料であるが，それを超えるとリクエストの使用量に応じて，料金が加算されてしまう従量課金制に変更された．リクエスト数を削減することにより，こういった API のコスト変化の面も柔軟に対応できると考える．

以下，本稿の構成を述べる．2 章では，関連研究を述べる．3 章では，提案手法を用いる際の前提知識を述べる．4 章では提案手法を述べる．5 章では，提案手法と網羅的な手法を用いた実験結果を述べ，6 章で，両手法を比較し考察を述べる．7 章で本研究のまとめと今後の展望を述べる．

2 関連研究

本章では，関連研究について述べる．我々が調査した限りロードマップを用いて，クエリ回数の削減を目的とした効率的なクローリング手法の提案を行った研究はほとんどない．しかし，本研究で用いる道路に関する研究は数多くあるのでそれらを解説する．また，本研究では提案手法と比較する手法としてメッシュを用いたクローリング手法を採用している．ジオソーシャルデータを扱った研究では，メッシュを用いて行う研究も数多くされているのでそれらについても説明を行う．

2.1 道路に関する研究

西村 [2] らは，OpenStreetMap から抽出したロードマップを用いて，道路の特性を推定する手法を提案した．まず，Point of Interest(POI) のみでの特性を推定する．推定した POI 付近の近隣道路の特性を推定することによって，ユーザの「にぎやかな道」を通りたいなどといったニーズに応えている．

余 [3] らは，不完全な道路ネットワークを考慮したマップマッチングを行いながら道路ネットワークを補間する手法を提案した．道路ネットワーク上に補完されない GPS 点である off-road にマップマッチングを行い，簡単化を行う．簡単化したオフロードに対して，DBSCAN を用いてクラスタリングを行い，道路の補間を行った．

竹内 [4] らは，道路ネットワークを用いて，略地図を生成するときに，経路の見やすさに着目している．具体的には，道路ネットワーク全体に経路の見やすさを考慮したコスト関数を用いることで，略地図を生成している．

2.2 メッシュを用いた研究

大森 [5] らは，Flickr 上にあるジオタグが付与された写真を多くクロールし，クロールしたデータのジオタグから海岸線などの地形を表す線を抽出するアルゴリズムの提案を行った．そのアルゴリズムでは，範囲をメッシュに区切り，クロールした写真データから地形を示す線を生成している．

Zhao [6] らは，効率的にジオソーシャルデータをクラスタリングする手法としてメッシュを用いてクラスタリングを行っている．具体的には，メッシュに区切った後，それぞれのメッシュに存在しているデータ数に基づいた番号を付けクラスタリングを行った．そのアルゴリズムは k-means と DBSCAN の両方の利点があることが分かった．

また，山田 [7] らは，メタデータが付与された風景写真を効果的に閲覧できるようにするための世界地図インタフェースシステムを実装した．世界地図上にメタデータに含まれる位置情報をもとに集めた写真を可視化するとき，多くの写真が撮影された場所には写真が密集してしまい，視認性が落ちてしまう．そこで，画面のズームレベルの大小によって，メッシュの大きさが変わるような実装を行い，視覚的に見やすい世界地図インタフェースの開発を実現した．

本研究は，メッシュに頼ることなくデータのクロールを行う点で上記の研究と異なる．また，地図上に無秩序に分布しているデータに対して，いかにクエリ回数を削減させ，効率良くデータをクロールできるかという点に重点を置いているところに本研究の意義があると考える．

3 準備

本章では，提案手法の前提知識を述べる．3.1 節で OpenStreetMap から抽出できるロードマップについて述べ，3.2 節では提案手法で用いる点の間引きを行うアルゴリズムの Ramer-Douglas-Peucker Algorithm について述べる．

3.1 OpenStreetMap から抽出できるロードマップ

図 1 に OpenStreetMap からクロールしたある範囲内のシャンゼリゼ通りと名前がつけられた道路を可視化した図を示す．

図 1 ロードマップ模式図

OpenStreetMap からクロールできるロードマップデータは GeoJSON ファイルとして保存することができ，そのファイル内にロードマップを構成している位置情報が含まれる．図 1 のように，ある範囲内の同じ名前の道路を取り出したときに，そのデータは両端の端点のみの位置情報で構成されているのではなく，短い道路が複数繋がって構成されていることが多い．また，それぞれの短い道路内には，図 1 の赤色のマーカーで示されているような交差点などの単一の位置情報を含んでいる．

このように，クロールしたそのままのロードマップデータに含まれている位置情報を何も操作を加えずにすべて用いて，クローリングを行ってしまうと，クエリ回数が増加し，リクエスト制限にかかる可能性がより高くなってしまうことが考えられる．そのため，提案手法では，クロールしたロードマップデータから端

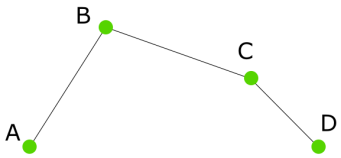

図 2　初期状態

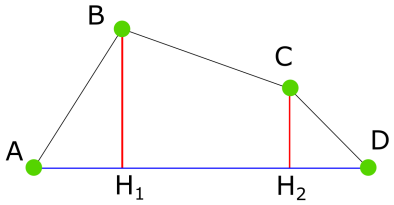

図 3　許容距離 ε との大小比較

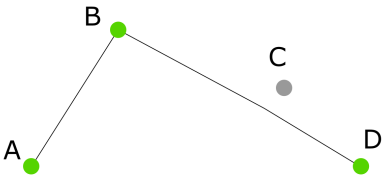

図 4　終了状態

点のみを抽出する．その際，次に述べる Ramer-Douglas-Peucker Algorithm を用いる．

3.2　Ramer-Douglas-Peucker Algorithm

Ramer-Douglas-Peucker Algorithm は，Ramer [8] が考案したアルゴリズムである．大量の点を結んで線を描くときにすべての点を用いて線を描くのではなく，それぞれの点から近くにある点を間引いて，元の線と近似させるアルゴリズムである．図 2，3，4 に，二次元平面上の点 A~D に Ramer-Douglas-Peucker Algorithm を適応する一連の操作の流れを示す．

まず，許容距離 ε を決める．始点 A，終点 D を結んだ直線 AD から残っているそれぞれの点 B，C までの距離を求め，許容距離 ε と比較し，間引く点を決定する．図 3 の直線 BH_1，CH_2 のような，残ったそれぞれの点と始点 および終点を結んだ直線 (直線 AD) との距離が許容距離 ε より大きかったら，その点は間引かず，小さかった場合はその点を間引く．図 4 では，直線 CH_2 が許容距離 ε より小さかったので，点 C が間引かれていることが分かる．これらの操作を再帰的に繰り返し，点を間引く．

このように，Ramer-Douglas-Peucker Algorithm は許容距離 ε によって，間引く点を決定する．したがって，許容距離 ε の大小に伴って，間引かれる点が変化する．許容距離 ε を 0 に近づければ，間引きが行われず，極端に大きくすると，ほぼすべての点が間引かれ，始点，終点のみしか残らない可能性が高くなる．

本手法では，OpenStreetMap からクロールしたロードマップデータにこのようなアルゴリズムを適応して，位置情報を間引き，クローリングを行いやすくする．

4　提案手法

本章では，提案手法を述べる．まず，はじめに，図 5 に提案手法の概要図を示す．本手法はロードマップデータを OpenStreetMap からクロールし，ロードマップの端点のみを抽出し，単一方向の道路に分類するフェーズ (ロードマップ抽出フェーズ) と実際にクエリ中心点を決定するフェーズ (クエリフェーズ) の大きく二つに分かれる．4.1 節では，ロードマップ抽出フェーズを述べ，4.2 節では，クエリフェーズを述べる．

4.1　ロードマップ抽出フェーズ

ロードマップ抽出フェーズの大まかな手順を下に示す．

1. ロードマップデータをクロール
2. 道路を名前ごとにマージ
3. 端点以外の位置情報を間引く
4. 道路の角度から東西南北判定を行う

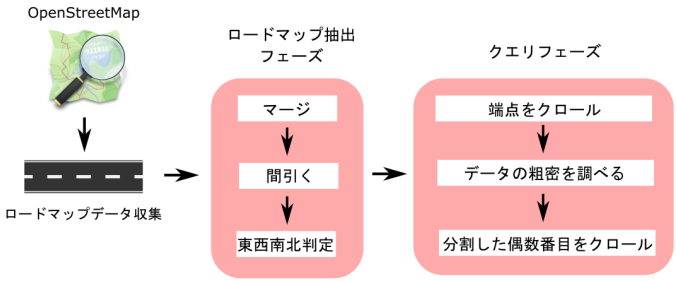

図 5　提案手法の概要図

まず，はじめに overpass turbo[4] を用いて，ロードマップデータのクロールを行う．OpenStreetMap のデータにアクセスできる API として，Overpass API[5] がある．overpass turbo は Overpass API を Web 上で利用できるデータマイニングツールである．OpenStreetMap 上に登録されている道路や建物の地物にはすべてタグ付けがされている．タグは施設や道路といった大まかな建物の種類を表すキーとレストランやカフェといった細かい建物の種類を表す値で構成されている．建物はただ単に大通りだけでなく，大通りからは外れているような住宅地に繋がっているような小道にも密集している可能性が高い．そのため，本手法では，大通りと細かい道路を示すタグを用いて，ロードマップデータをクロールする．具体的には，表 1 に示したタグを用いてロードマップデータをクロールする．

表 1　本手法で用いる道路タグ

キー	値
highway	primary secondary tertiary unclassified residential pedestrian

クロールできたロードマップデータには，その道路の名前がデータとして付与されている．同じ名前が付与されている道路は複数あることもある．そこで，複数の同じ名前の道路をひとつの

[4] http://overpass-turbo.eu/
[5] http://overpass-api.de/

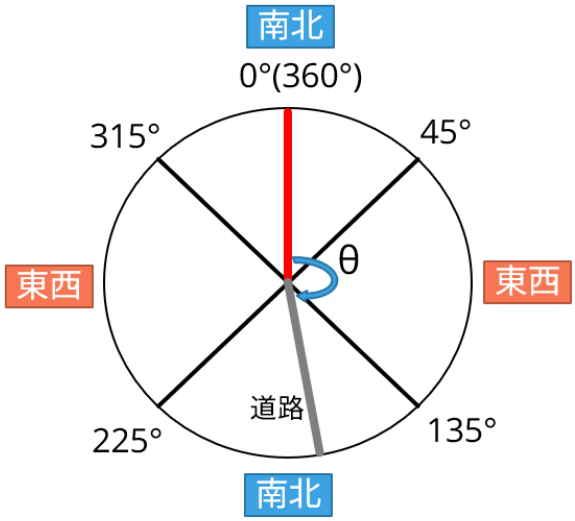

図 6　東西南北判定

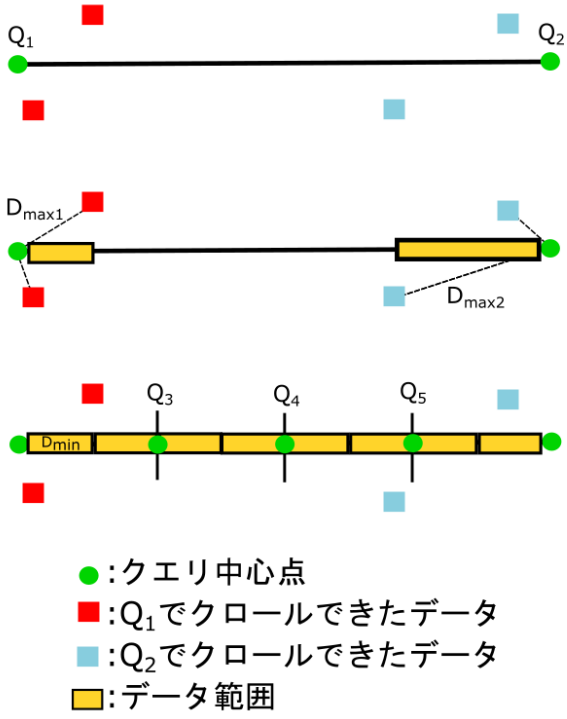

図 7　クエリフェーズ

道路にマージする．

マージできたそれぞれの道路には，端点以外に複数の単一の交差点などを示す位置情報が含まれている．端点のみを取り出したいので，先に述べた Ramer-Douglas-Peucker Algorithm を用いて，単一の位置情報を間引きし，端点のみの位置情報を抽出する．

図 6 に，東西南北判定を行うイメージ図を示す．端点のみの位置情報を抽出できたすべての道路に対して，その角度を求める．図 6 のように，北向き 0°の線と道路がなす角度　θ を求め，その角度に応じて，道路を南北方向，東西方向の 2 パターンの道路に分類する．具体的には，角度が $0° \leq \theta \leq 45°$，$135° \leq \theta \leq 225°$，$315° \leq \theta \leq 360°$ なら南北方向の道路，$45° \leq \theta \leq 135°$，$225° \leq \theta \leq 315°$ なら東西方向の道路として判定する．

端点のみを取り出し，かつ南北もしくは東西のどちらかの方向に分類した道路を用いて，実際にクエリの中心点を決定するクエリフェーズを行う．

4.2　クエリフェーズ

クエリフェーズの大まかな手順を下に示す．図 7 にクエリフェーズの模式図を示す．本手法のクエリフェーズは指定したクエリ中心点からジオソーシャルデータを近さ順にクロールする場合のみに焦点を当てている．

1. 道路の端点 (Q_1，Q_2) を用いて，クロールを二回行う
2. 端点 (Q_1，Q_2) とそれぞれの端点でクロールできたデータとの距離の最大値 (D_{max1}，D_{max2}) を求める
3. D_{max1}，D_{max2} のうち，小さいほうを D_{min} とし，道路を等分割する
4. 分割した道路の始点と終点を除く偶数番目をクロールする

まず，はじめに，直線道路の両端の端点である二点 (Q_1，Q_2) をクエリ中心点としてクロールを二回行う．

Q_1，Q_2 のクロールを行ったあと，端点 Q_1，Q_2 とそれぞれの端点でクロールできたデータとの距離をすべて求める．求めた距離のうち，その最大値をそれぞれ D_{max1}，D_{max2} とする．

最大値を求めることによって，用いた道路に対して，それぞれの端点 Q_1，Q_2 のクローリングから収集できたおおよそのデータの範囲を知ることができる．データの散らばりが密であるほど，端点から最も離れているデータより内側に新しいデータがある可能性が高い．Q_1，Q_2 のクローリングで集めることができなかったデータを集めるために，次の操作を行う．

求めた D_{max1}，D_{max2} のうち，小さいほう（データが密に集まっているほう）の距離を D_{min} として，道路を等分割する．具体的には，用いた道路の距離を D_{min} で割る．割った値が小数の場合は，切り上げを行い，奇数の場合は 1 を足して，値を必ず偶数にする．

分割できた道路の始点と終点 (Q_1，Q_2) を除く，偶数番目をそれぞれクロールする．

ここで，例外として図 8 を示す．それは複数の名前の異なる道路が同じ共有点で接続している場合である．接続している複数の道路は共通の端点を接続点として共有しているので本手法のクエリフェーズを適応して，クロールを行うと接続点を複数回クロールしてしまうことになる．そのため，このような場合，接続点は一度クロールしたら二度とクロールを行わないものとする．

5　実験

本章では，提案手法とメッシュを用いて，網羅的にデータをクロールする手法 (ベースライン手法) を用いて行った実験について述べる．5.1 節では，データのクロールを行う際の条件とベー

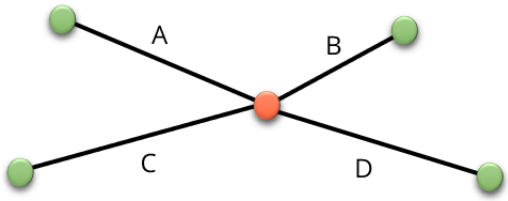

図 8　複数の道路が接続している場合

スライン手法について述べる．5.2 節では両手法を用いての実験を実現するために行った実装について述べ，5.3 節で実験結果を述べる．

5.1　クロール条件とベースライン手法

今回の実験では Google Places API に含まれる機能である Nearby Search を用いてデータをクロールする．Nearby Search は，1 クエリで指定した緯度経度 (クエリ中心点) から近さ順で建物のデータを 20 件分クロールすることができる機能である．データを近さ順でクロールすることができるので，本手法のクエリフェーズが適応できる．

Nearby Search を用いるときに必須のパラメータとして建物の種類 (タイプ) を設定しなければならない．タイプはカフェ (cafe) やホテル (lodging) など約 100 種類が提供されており，今回の実験では，レストラン (restaurant) をタイプとして設定する．データをクロールする範囲として，以下の二つの範囲を用いる．一つは，パリの凱旋門から伸びるシャンゼリゼ通りを含んでいる範囲である (範囲 1)．もう一つは，スペインのサンセバスティアンと呼ばれる地域を含んでいる範囲である (範囲 2)．どちらの範囲とも，レストランが密集している地域であり，多くのデータをクロールすることができると考えられる．

ロードマップ抽出フェーズでは，道路を南北か東西かの道路に分類し，どちら一方の道路のみを用いて，それぞれクエリを行う．今回の実験では，それぞれの範囲に対して，南北方向の道路を用いた場合，東西方向の道路を用いた場合の両方のパターンで実験を行った．

また，クエリ中心点が範囲の境界付近に設定する場合，範囲外のデータをクロールすることも考えられる．このとき，範囲外に存在しているデータをクロールしてしまった場合，それらのデータも実験結果には含めるものとする．

ベースライン手法では，範囲内をメッシュに区切り，データを網羅的に集めることを考える．範囲内を任意の個数のメッシュに区切り，区切られたそれぞれのメッシュの中心点をクエリ中心点として，クロールを行う．範囲 1 では，範囲内を 225 個のメッシュに区切り，合計 225 回のクロールを行った．範囲 2 では，範囲内を 100 個のメッシュの区切り，合計 100 回のクロールを行った．

5.2　実装

ロードマップの位置情報の間引きでは，Python3 の点の間引きを行うライブラリである rdp[6] を用いて，Ramer-Douglas-Peucker Algorithm を適応した．今回用いたライブラリ rdp は緯度経度を直接用いたユークリッド距離で距離計算を行う．ロードマップ抽出フェーズ内で，位置情報を間引くときは，最低限道路の端点を残しつつ，その間に含まれている単一の位置情報を間引ければよい．そのため，今回の実験では，両方のクロール範囲とも間引きで用いる許容距離 (ε) を 0.0001 に設定した．

5.3　実験結果

表 2，3 に範囲 1 で東西方向の道路および南北方向の道路を用いた場合の実験結果を示す．表 4，5 に範囲 2 で東西方向の道路および南北方向の道路を用いた場合の実験結果を示す．合計クエリ回数 Q_{all} は一連のクローリングでクエリを行った合計回数である．総データ数 D_{all} はクロールできたすべてのデータ数である．純データ数 D_{unq} はクロールできた重複を含まない固有のデータ数であり，重複データ数 D_{dup} は総データ数 D_{all} から純データ数 D_{unq} を引いたデータ数である．積集合データ数 D_{insec} はどちらの手法でもクロールできた純データ数であり，差集合データ数 D_{diff} はどちらか一方の手法でのみクロールできた純データ数である．

6　考察

まず，範囲 1 における東西方向の道路と南北方向の道路を用いた場合の実験結果について，考察する．

提案手法で東西方向の道路を用いた場合，ベースライン手法よりも多くの純データをクロールすることに成功し，クエリ回数も約 5 割削減することに成功した．この要因は，範囲 1 での東西方向の道路の本数が単純に多かったことやその道路が比較的，大きい道路であったことが挙げられる．建物が立っていやすいと考えられる大きい道路沿いにクロールを行い，網羅的にデータを集めることができたと考えられる．

一方，南北方向の道路を用いた場合，ベースライン手法でのクエリ回数を約 7 割削減することに成功したが，ベースライン手法より多くの純データをクロールすることができなかった．本手法でのクローリングはクエリに用いる道路の本数に比例する．南北方向の道路は東西方向の道路より数が少なかったことにより，クエリ回数を削減することに成功したがクエリ回数の減少により，ベースライン手法を超える純データをクロールできなかったことが考えられる．

次に，範囲 2 での結果について考察する．東西方向の道路を用いた場合，クエリ回数を約 4 割削減しつつ，ベースライン手法より多い純データをクロールできた．同様に，南北方向の道路でも，約 6 割のクエリを削減できた．範囲 2 は，道路が格子状に交わっている範囲に多くのレストランが密集している地域である．格子状に交わっている道路の東西，南北のどちらかの方向の道路を用いたことがクエリ数を重ねることなく，ベースライン手法よりも多い純データをクロールできたことにつながったと考える．

[6] https://pypi.org/project/rdp/

表 2　範囲 1 の実験結果 (東西方向)

	提案手法	ベースライン手法
合計クエリ回数 Q_{all}	121	225
総データ数 D_{all}	2420	4500
純データ数 D_{unq}	746	602
重複データ数 D_{dup}	1674	4438
積集合データ数 D_{insec}	580	580
差集合データ数 D_{diff}	166	22

表 3　範囲 1 の実験結果 (南北方向)

	提案手法	ベースライン手法
合計クエリ回数 Q_{all}	65	225
総データ数 D_{all}	1300	4500
純データ数 D_{unq}	588	602
重複データ数 D_{dup}	712	4438
積集合データ数 D_{insec}	475	475
差集合データ数 D_{diff}	113	127

表 4　範囲 2 の実験結果 (東西方向)

	提案手法	ベースライン手法
合計クエリ回数 Q_{all}	66	100
総データ数 D_{all}	1320	2000
純データ数 D_{unq}	174	166
重複データ数 D_{dup}	1146	1834
積集合データ数 D_{insec}	165	165
差集合データ数 D_{diff}	9	1

表 5　範囲 2 の実験結果 (南北方向)

	提案手法	ベースライン手法
合計クエリ回数 Q_{all}	44	100
総データ数 D_{all}	880	2000
純データ数 D_{unq}	180	166
重複データ数 D_{dup}	700	1834
積集合データ数 D_{insec}	157	165
差集合データ数 D_{diff}	23	9

両方の範囲において，本手法を用いて，クエリ回数を削減できたので，削減できた分のクエリを他の範囲のクローリングに充てることで，合計的により少ないクエリ数で複数の範囲のクローリングを行え，より短時間で多くのデータをクロールすることができると考える．

7　おわりに

本研究では，建物は道路に沿って建てられているという仮説のもと，OpenStreetMap から抽出できるロードマップを用いて，効率的なジオソーシャルデータのクロールを試みた．実験の結果，提案手法を用いて，合計クエリ回数を，範囲 1 の東西方向の道路のクローリングでは約 5 割，南北方向の道路では約 7 割，範囲 2 では，東西方向の道路では約 4 割，南北方向の道路では約 6 割削減することに成功した．ロードマップを用いた本手法では，ベースライン手法より少ないクエリ回数で各範囲のデータを網羅的にクロールすることができたうえにベースライン手法よりも多い純データをクロールすることができたので，ジオソーシャルデータをクロールする際に，ロードマップは有用であることを示せた．

今後の課題として，さまざまな道路に対応することやクエリ回数をより削減できるようにクエリフェーズの見直しを行うことを検討している．さらに，重複データを少なくすることを検討している．また，本手法では，道路を名前ごとにマージしたが，日本のようにひとつひとつの道路に名前がつけられていない場合も多い．そのような地域にも対応できるように生のロードマップデータをどのように扱うのかも検討する必要がある．

今回の実験では，範囲内を網羅的にクローリングできるはずのベースライン手法はメッシュの一辺の大きさを特に指定せず，メッシュの個数のみを指定して，クローリングを行った．そうではなく，各メッシュの一辺の長さをより細かく調整し，比較手法として適切にベースライン手法を再構築する必要もある．

また，クエリ回数やクロールできたデータ数だけで結果を比較するのではなく，クロールできたデータの質も考慮したうえでデータの考察を行うことも考えている．

謝辞

本研究は JSPS 科研費 JP19K11982 の支援の助成を受けたものです．ここに記して謝意を表します．

参考文献

[1] 200 の国と地域を網羅する google マップのおもしろ統計｜@dime アットダイム. https://dime.jp/genre/499471/.
[2] 西村拓哉, 西田京介, 戸田浩之, 澤田宏. ソーシャルメディアを用いた道路の特性理解. 第 8 回データ工学と情報マネジメントに関するフォーラム (DEIM Forum 2016), pp. 1–8, 2016.
[3] 余家豪, 佐々木勇和, 石川佳治. 不完全な道路ネットワークにおけるマップマッチングとクラスタリング手法を用いた道路セグメントの補間手法の提案. 第 9 回データ工学と情報マネジメントに関するフォーラム (DEIM Forum 2017), pp. 1–6, 2017.
[4] 竹内真樹, 森口昌樹, 今井桂子. 指定された 2 点間の経路の見やすさを考慮した略地図生成. 情報処理学会第 75 回全国大会, Vol. 6, p. 6, 2013.
[5] 大森雅己, 廣田雅春, 石川博, 横山昌平. ソーシャルメディア上から収集したジオタグに基づく地理的特徴の抽出と評価. 情報処理学会論文誌データベース (TOD), Vol. 8, No. 1, pp. 1–16, 2015.
[6] Qinpei Zhao, Yang Shi, Qin Liu, and Pasi Fränti. A grid-growing clustering algorithm for geo-spatial data. *Pattern Recognition Letters*, Vol. 53, pp. 77–84, 2014.
[7] 山田匠, 廣田雅春, 石川博, 横山昌平. メタデータを利用した風景写真の時系列変化を効果的に閲覧するための世界地図インタフェース. 第 7 回データ工学と情報マネジメントに関するフォーラム (DEIM Forum 2015), pp. 1–7, 2015.
[8] Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer graphics and image processing*, Vol. 1, No. 3, pp. 244–256, 1972.