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

余　家豪† 佐々木勇和†† 石川　佳治†

† 名古屋大学大学院情報科学研究科　〒 464–8601 愛知県名古屋市千種区不老町
†† 大阪大学大学院情報科学研究科　〒 565–0871 大阪府吹田市山田丘 1 番 5 号
E-mail: †yu@db.ss.is.nagoya-u.ac.jp, ††sasaki@ist.osaka-u.ac.jp, †††ishikawa@is.nagoya-u.ac.jp

あらまし　近年，GPS を搭載しているデバイスが一般的となっている．GPS での位置測位では，誤差を含んでいることが多く，正確に位置を測定することは難しい．誤差を修正するための技術として，マップマッチングがある．多くのマップマッチングアルゴリズムでは，完全な道路ネットワーク（つまり，全ての道路が欠損していない道路ネットワーク）を前提としている．しかし，不完全な道路ネットワークも世の中には数多く存在している．不完全な道路ネットワークを想定した手法では，移動体が道路ネットワークから離れた場合に道路上以外を移動していると判断し，GPS データをそのまま出力する．このような手法では，GPS の位置を補正はするが，道路ネットワークを補間することは行わない．そこで，本研究では，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実行しながら，道路ネットワークを補間していくことを目的とする．実際の車両 GPS 軌跡データを用いて，提案手法が道路ネットワークを正しく補間できていることを確認する．

キーワード　不完全な道路ネットワーク, マップマッチング, クラスタリング,DBSCAN, 補間

1. はじめに

近年，カーナビゲーションシステムやスマートフォン等，GPS を搭載しているデバイスが一般的となっている．GPS での位置測位では，誤差を含んでいることが多く，正確に位置を測定することは難しい．誤差を修正するための技術として，マップマッチングがある [2] [3] [5]．マップマッチングは，基本的に車載 GPS を考慮し，自動車は道路上のみを移動するということを前提に GPS の位置情報を道路上に修正するという技術である．多くのマップマッチングアルゴリズムでは，完全な道路ネットワーク（つまり，全ての道路が欠損していない道路ネットワーク）を前提としている．しかし，不完全な道路ネットワークも世の中には数多く存在している．例えば，OpenStreetMap(注1) のような手動にて更新される道路図や，発展途上国の道路図などが考えられる．完全な道路ネットワークを前提とするマップマッチングアルゴリズムに対して，不完全な道路ネットワークを入力すると，大きな誤差を含んだ結果となってしまう．図 1 は，OpenStreetMap より抽出した道路ネットワーク上（黒色の線分）に北京のタクシーの GPS データ（赤色の点）をプロットしたものである [8] [9]．図の中心部分において，道路上を移動していない GPS データが存在する．地図データでは欠損しているが実際には存在する道路である．図 2 に，完全な道路ネットワークを前提としたマップマッチングアルゴリズムを適応した結果を示す．緑色の点がマップマッチングの結果を示すが道路ネットワークにない道路を通った場合，マップマッチングした軌跡が切断されたようなありえない結果が返ってくる．

一方，不完全な道路ネットワークを前提とするマップマッチングアルゴリズムも提案されている [4] [6]．不完全な道路ネットワークを想定したマップマッチング手法では，移動体が道路ネットワークから離れた場合に道路上以外を移動していると判断し，GPS データをそのまま出力する．このような手法では，GPS の位置を補正はするが，道路ネットワークを補間することは行わない．道路ネットワーク上に補正されない GPS 点は，off-road と呼ばれ，この点が不完全な道路ネットワークを考慮した手法の特徴である．図 3 は，不完全な道路ネットワークを考慮したマッチング結果を示す．図 2 の結果とは異なり，道路上を移動していないと判断された点をそのまま出力されることにより，自然な移動軌跡を出力できていることがわかる..

不完全な道路ネットワークを想定したマップマッチングでは，軌跡データを補正することはできるが，道路ネットワークそのものを補間することはできない．そこで，本研究では，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実行しながら，道路ネットワークを補間していくことを目的とする．実際の車両 GPS 軌跡データを用いて，提案手法が道路ネットワークを正しく補間できていることを確認する．単純に出力された off-road を道路として補間する場合，三つの問題がある．先ず，一つ目は，実際に道路が無いにも関わらず，GPS の誤差により off-road と判定される場合である．この場合，道路を補間することは不適切である．off-road 軌跡で道路かどうかを判定すると，この軌跡の連続な GPS 数が重要である．二つ目は，GPS データに誤差が含まれるため，そのまま GPS データを補間するだけでは，非現実的な形状で補間してしまうことである．そのため，現実的な形状に変換して補間を行う必要がある．三つ目は，複数の off-road が似たような位置に出力された場合であ

(注1)：https://www.openstreetmap.org/

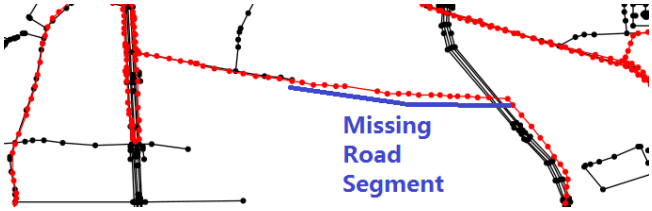


図 1: 不完全な道路ネットワーク上の GPS 軌跡

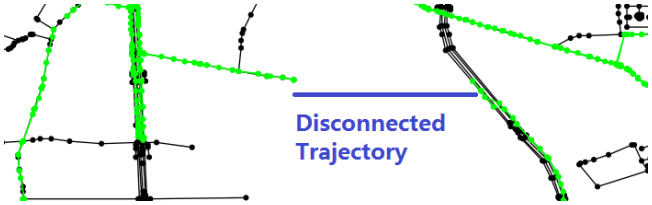


図 2: 完全な道路ネットワークを前提とするマップマッチングの結果

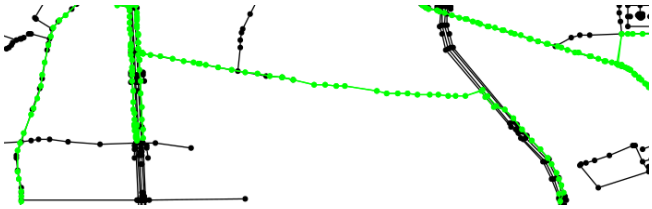


図 3: 不完全な道路ネットワークを考慮したマップマッチングの結果

る．この場合，同じ道路を表している可能性が高いため，統合して補間する必要がある．

提案手法は，まずマップマッチングアルゴリズムを用いて，十分な個数の連続の GPS 点がある off-road 軌跡を抜き出す．その後，off-road を簡単化により，現実的な形状に変換する．各 off-road 軌跡の位置情報を代表する中心点をデータセットして DBSCAN でクラスタリングする．クラスタ中の軌跡を組合せて道路ネットワークのサブグラフを構築し，最後に補間を行う．

本稿の構成は以下のとおりである，まず，完全な道路ネットワークを前提としたマップマッチング手法と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手法を 2. で論じる．3. には，事前準備である．我々の提案手法は 4. で説明し，5. で実験し，実験結果を示す．最後に，6. で本稿のまとめと今後の課題について述べる．

2. 関 連 研 究

本章では完全な道路ネットワーク，不完全な道路ネットワークにおけるマップマッチングおよび，道路ネットワークの補間手法に関する研究をそれぞれ紹介する．

完全な道路ネットワークにおけるマップマッチングでは，Lou らの ST-Matching アルゴリズム [2] と，Newson らの Hidden Markov Model に基づく手法 [3] がある．両方の手法とも，まず，GPS 点から道路ネットワーク上の複数個の点を候補点として抽出する．各 GPS 点の候補点を時間軸上に並べ，それらをつなぐことにより，候補点グラフ G_{cand} を構築する．二つのアルゴリズムでは，各候補点間の枝の重みが異なる．ST-Matching アルゴリズム [2] では，空間と時間の連続性をもとに枝の重みを決定する．Hidden Markov Model に基づく手法 [3] では，確率密度などをもとに枝に確率的な重みを与える．最後に，両手法とも候補点グラフにおける，経路上の重みが最も大きくなる一本の候補経路を取り出し，それをマッチした軌跡とする．これらの手法では，候補点グラフは，な道路上の点しか考慮していないため，GPS 点をそのまま出力することはい．

不完全な道路ネットワークを考慮した手法の一つ Pereira らの手法 [4] として, 遺伝的アルゴリズムを用いて，マッチング精度を改善している．しかし，遺伝的アルゴリズムを用いた手法は，NP 困難問題である．Pereira らの手法は最適な結果を得られることを保障できない．Hidden Markov Model に基づく手法を Jan-Henrik らが拡張している [6]. [6] のアイデアは，候補点グラフに，GPS 点をそのまま候補点として追加することである．候補点グラフ G_{cand} においては，枝の重みを計算する上で，三種類の要素: (1) GPS 点と候補点間のユークリッド距離, (2) 道路上の候補点における道路ネットワーク上での最短経路距離，および (3) 連続する 2 つの候補点が生の GPS 点か道路上に修正した候補点かにより場合分けした値を考える．三つの要素の値を掛け合わせるものを枝の重みとする．候補点グラフ G_{cand} を作成した後は，他の手法と同様に一本の候補経路を取り出し，マッチング結果とする．本稿では, [6] の手法を用いてマップマッチングを実行する．

道路ネットワークデータを用いずに GPS 軌跡データから道路図を作成することを目的とする手法として，論文 [1] で提案された Mininng Transfer Network がある．Chen らは，GPS 軌跡データにより Tranfer Network を得るために，Coherence Expanding アルゴリズムを提案した．軌跡の交差点または端点を頂点として連接する．このように，Transfer Network が作成される．しかし，この手法を利用して作成された Tranfer Network は，実際の道路ネットワークに対して誤差が大きくなる．

3. 準　　備

本稿では，オフラインのマップマッチングを想定する．つまり，GPS データの軌跡は始点から終点まで既に全て揃っており，新たな GPS 点が追加されることはない．GPS 点は，誤差を含む緯度・経度の情報，測定された時間情報，および機器の識別子から成っている．どの程度の誤差を含むかは自明ではないとする．GPS 点の集合を識別子毎に時間軸順に並べることにより，軌跡を求めることが可能である．道路ネットワークは，節点と枝から構成され，節点は緯度・経度情報を保持している．そのため，GPS 点が道路ネットワーク上の節点もしくは枝からどの程度離れているかを計算することが可能である．この道路ネットワークは，不完全な道路ネットワークであり，一部が欠損していることを想定する．ただし，完全な道路ネットワークの場合でも動作し，この場合マップマッチング実行するだけとなる．

3.1 表　　記

本論文では道路ネットワークを有向グラフ $G = (V, E)$ で表

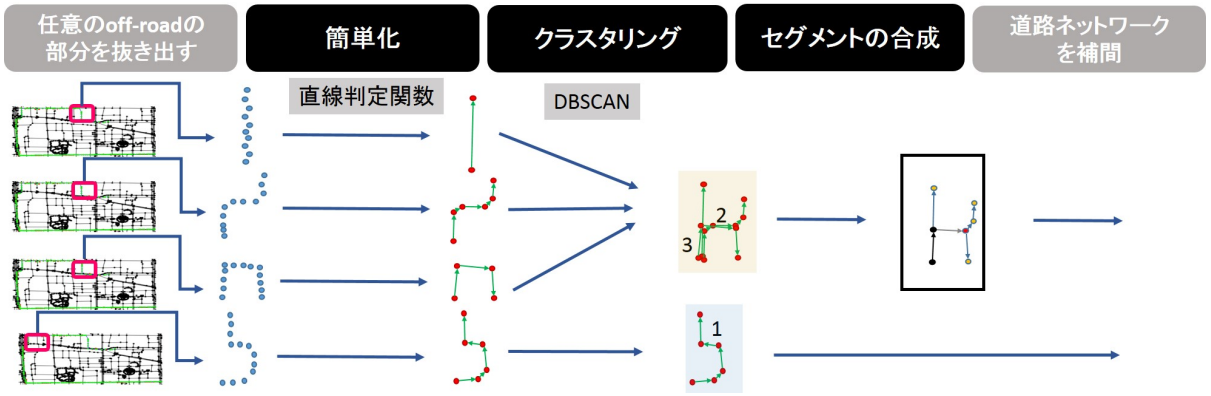


図 4: 提案手法の概観

す．V は頂点の集合であり，E は辺の集合である．タクシー軌跡を $T = \{pt_1, \cdots, pt_n\}$ と書く．グラフ G 上の任意 2 点 $u, v \in V$ 間の最短路距離を $dist_G(u, v)$ と書き，u から v へ経路が存在しない場合は $dist_G(u, v) = \infty$ とする．また，任意の二点 s, t 間のユークリッド距離は $len(s, t)$ で表す．

3.2 マップマッチング

本稿では，不完全な道路ネットワークを想定したマップマッチング手法 [6] に基づいて，道路ネットワークを補間する．この手法では，まず，任意の $pt_i \in T$ に対して，マッチングの候補点の集合 $C_i = c_i^0, ..., c_i^K$ を用意する．K はユーザが設定するパラメータであり，候補点集合の最大サイズを表す．c_i^0 は，pt_i の位置情報を持つ（つまり，GPS と同じ位置情報を持つ）特殊な候補点である．まず，この手法では，2 つの連続な点である pt_{i-1} と pt_i に対応する候補点に対して，ユークリッド距離と道路ネットワーク上の最短距離を求める．これらの値はどちらも近い方が候補点間の遷移の確率が高くなる．さらに，この手法では，候補点が GPS 点がどうかも遷移確率の計算に用いる．最終的に pt_1 の候補点から pt_n の候補点までの遷移において，確率が最も高い経路をマップマッチングの結果とする．

4. 提 案 手 法

本手法では, [6] の手法をもとに off-road の GPS 点から道路ネットワークを補間する．図 4 に提案手法の概要を示す．提案手法では，マップマッチングにて抽出された off-road に対して，簡単化を行う．これにより，直線もしくは折れ線を取得することができる．次に，簡単化した off-road を DBSCAN[8] によりクラスタリングを実行する．クラスタリングにより同一区域の off-road と判定されたものを統合する．その後，道路ネットワークを補間する．

4.1 簡 単 化

簡単化は, [1] で提案されている Mininng Transfer Network を改良して行う．Mininng Transfer Network では，地図データを用いずに GPS 軌跡データから道路図を作成することを目的としている．一方で，提案手法では，連続的な GPS 点を最低限の点数で折線近似を行う．ユーザは off-road 軌跡の GPS の最小点数 τ をパラメータとして設定する．τ 未満の off-road は補間の対象外とする．簡単化では，まず直線判定を行う．始点 pt_1 と終点 pt_n を基準として，全ての点の角度, と pt_1 と pt_n が成す線分までの距離を求め，角度が θ 以下かつ距離が δ 以下なら直線と判定する．終点から順に削除していき，直線と判定するまで反復して計算する．次に，削除された点に関して，同様に直線判定を行い，最終的にすべての点をつなげることにより，折線を作成する．簡単化の具体的な計算方法をアルゴリズム 1 に示す．

4.2 クラスタリング：DBSCAN

DBSCAN は密度をベースとしてクラスタリングの代表的な手法である．DBSCAN では，パラメータとして最小点数 $MinPts$ と半径 ϵ を与え，ある点を中心とした半径 ϵ 内に $Minpts$ 以上の点があれば，半径内の点を同じクラスタと判断する．提案手法では，簡単化した off-road を DBSCAN によりクラスタリン

Algorithm 1 Simplication

Input: off-road 軌跡 $T_{off} = \{pt_1, \cdots, pt_n\} (n \geqq \tau)$
Output: Simplied Sequence T_{result}

```
function SIMPLIFY(T_off)
    T_result ← T_off.begin()
    while T_off.size ≧ 2 do
        T_backup ← T_off
        while !straight_line(T_backup) do        ▷ 直線判定関数
            T_backup.pop();    ▷ 直線判定の条件を満たさない点を削除
        end while
        T_result ← T_result ∪ T_backup.end();  ▷ 直線判定の条件を満たす点を結果に入れる
        T_off.remove(T_backup.begin(), T_backup.end() − 1);    ▷ off-road 軌跡を更新
    end while
    return T_result
end function
```

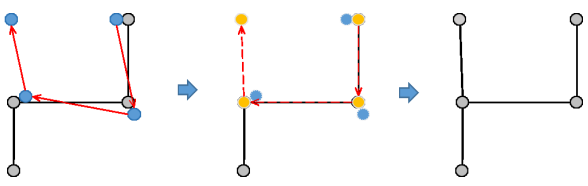


図 5: サブグラフにマップマッチング

グを行い，接続している off-road を抽出する．まず，簡単化された off-road の中心点を求め，この中心点を基に DBSCAN を実施する．中心点は，各点の座標の平均点とする．一番長い中心点から各点までの距離を最小半径 r とする．提案手法では，パラメータ ϵ を全ての off-road の最小半径 r のうち，最大のものとする．一方で，$MinPts$ は，道路の補間の信頼度に大きく関わる．この値をユーザが大きく設定すれば，より正確な道路の補間を実現できるが，補間する道路の数は少なる．

4.3 セグメントの組合せ

クラスタリングにより同一区域内の道路と判定された off-road 軌跡に対し，セグメントを組み合せることで最終的に補間する道路を求める．同じ区域に存在する off-road 軌跡でも，その幾何的な情報は異なる可能性がある．例えば，住宅街にあるような小道を考える．大通りの道路とは異なり，そうした小道の場合いくつかの道路へと複雑につながっていることが多い．このような道路上を自動車が通った場合，得られる GPS 軌跡もどの道からどの道へ抜けたかによって異なってくる．したがって，複数の軌跡データを統合し，その区域の道路ネットワークを生成する必要がある．

セグメントの組合せは，各 off-road 軌跡をグラフとみなし，統合することで実現する．まず，クラスタ内で最も頂点数が多い簡単化された off-road 軌跡を抽出し，サブグラフ $G_{sub} = (E', V')$ とする．つまり，off-road 軌跡中の各点を頂点集合 V'，各点間の遷移を辺集合 E' とみなす．次に，クラスタ内の他の off-road 軌跡を一つずつサブグラフに統合する．具体的には，まず off-road 軌跡中の任意の点について，ユーザによって定められた半径 σ 以内で一番近い頂点 $v \in V'$ を選択し統合する．もし半径 σ に頂点が存在しない場合は，半径 σ 以内の辺に節点を作成し，その点を新たな頂点として頂点集合 V' に加える．つまり，この場合では辺上に新たな頂点が作成されるため，道路ネットワークとしては交差点が生成されることになる．もし半径 σ 以内に頂点も辺も存在しない場合は，その点を新たな頂点 v_{new} として V' に追加する．頂点の追加が終わったら，off-road 軌跡中の遷移を対応する頂点間の辺として辺集合 E' に追加する．以上の処理をクラスタ中の全ての off-road 軌跡に対して行うことで，統合されたサブグラフ G_{sub} が得られる．

図 4.3 で具体例を用いて説明する．まず，クラスタ内の off-road 軌跡のうち最も頂点数が多いものをサブグラフとする．距離が σ 以内の点に関しては統合し，それ以外のものは新たな節点とする．これにより，補間する道路ネットワークを作成することできる．

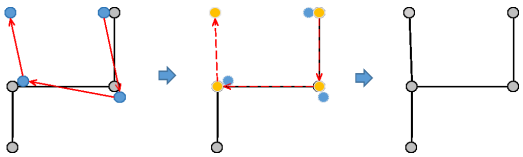


図 6: サブグラフにマップマッチング

これまでの処理により off-road 軌跡から欠損道路を表すサブグラフが生成できたため，これを元の道路ネットワークと結合する．一般に，このようなグラフの結合はどのように行うかが問題となるが，本手法では off-road の特徴により結合を容易に実行できる．**??**項で述べたとおり，off-road 軌跡の少なくとも片方の末端は道路ネットワーク上に存在する．したがって，そのような末端を道路ネットワークと自然に接続するだけで，道路ネットワークとの結合が実現できる．

以下，具体的な処理の手順を述べる．なお，図 7 は実際に，サブグラフ G_{sub} を道路ネットワーク G に結合する際の様子を示す．まず，サブグラフ中の頂点 $v' \in V'$ に関して，道路ネットワーク上に存在するものを V に加える．このとき，必ずしも v' が道路ネットワーク上の頂点と一致しないことに注意する．つまり，道路ネットワークの辺 $e = (a, b) \in E$ 上に v' が配置される可能性もある．なお，a, b はそれぞれ辺 e の始点と終点を表す．このような場合，一旦辺 e を辺集合 E から削除して，挿入した点 v' を考慮した辺 $(a, v'), (v', b)$ を追加する必要がある．サブグラフの中で道路ネットワーク上に存在する点を全て追加し終えたら，次は残りの off-road 上の点と辺を追加し，道路ネットワークの補間を完了する．

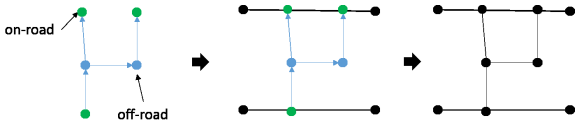


図 7: 道路ネットワークに結合

5. 実　　験

本章では，実データを用いて，提案手法の補間精度と効率性を検証する．補間精度に関しては，直線・折線・複数の折線という三つの場合の道路ネットワークの補間結果を評価する．なお，不完全な道路ネットワークとしては OpenStreetMap のデータを用い，補間結果の確認には対象領域の Google Map を用いた．一方で，提案手法の効率性については，パラメータ $MinPts$ を変化させた際の生成されたクラスタの生成数と補間結果の正確性を評価した．

表 1: タクシーデータ

タクシー台数	101
最小 GPS 点数	6746
最大 GPS 点数	147739
平均 GPS 点数	30617

表 2: 道路ネットワーク

取得日時	2015 年 6 月 27 日
位置	北京市
節点数	306311
枝数	334315

5.1 実験データ

軌跡データセットとしては，北京市内のタクシーから得られた GPS の軌跡データを用いる [8] [9]．このデータセットには 2008 年 2 月 2 日から 8 日まで北京市内を走行した 10,357 台のタクシーのデータが含まれており，タクシーの識別子，緯度経度とタイムスタンプから構成される．しかし，経緯度データやタイムスタンプデータの欠損，連続的なはずの二つの GPS 点の距離が異常に離れていること，タイムスタンプの前後関係の間違いなど，いくつかの問題も含んでいる．そのため，前処理

図 8: 直線の場合

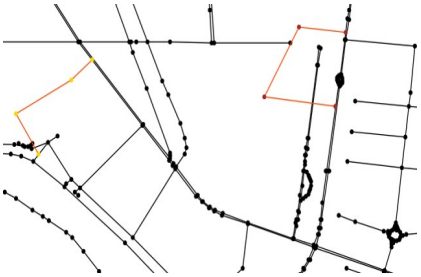


図 9: 折線の場合

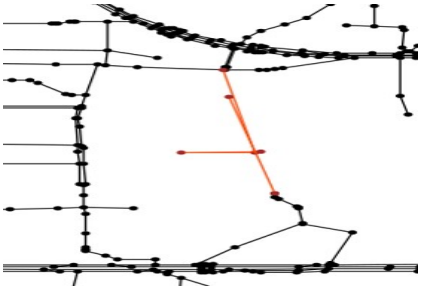


図 10: 複数の折線の場合

図 11: 図 8 の対象領域の Google map

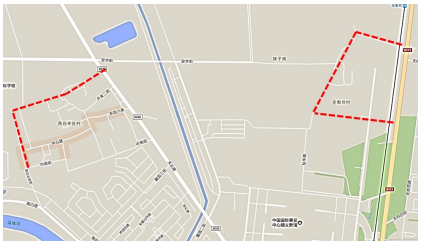


図 12: 図 9 の対象領域の Google map

図 13: 図 10 の対象領域の Google map

としてこれらの異常となるデータを削除する．また，道路ネットワークの範囲外となるデータや GPS 点の数が少なすぎるタクシーのデータも除去した．表 1 にデータセットの中身を示す．

道路ネットワークとしては，OpenStreetMap から北京市内のデータを抽出した．表 2 に道路ネットワークのデータを示す．

5.2 実験環境

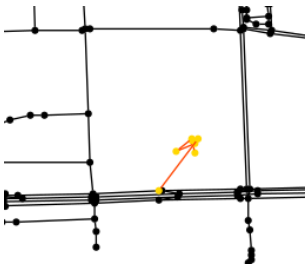
提案手法を C++11 を用いて実装し，gcc 5.4.0 を用いて最適化オプション −O3 の設定でコンパイルした．本実験は Ubuntu 16.04.1 LTS(CPU:Intel(R) Core(TM) i7 CPU 870 @ 2.93GHz, メモリ：8GB) で行った．

5.3 補間の具体例

本実験では，まず OpenStreetMap の道路ネットワークとタクシーデータに対して，提案手法を用いて道路ネットワークの補間を行う．次に，道路ネットワークの補間の結果が実際の道路として存在するかどうかについて，Google Map を用いて検証する．ここでは，直線・折線・複数の折線という三種類の道路ネットワークの補間結果を示す．なお，クラスタリング手法で用いたパラメータ τ と $MinPts$ はそれぞれ 5 と 2 に設定した．また，直線判定関数の中で用いるパラメータは $\theta = 15$，$\delta = 20$ に設定した

実験結果を図 8-9 に示す．図 8 では，直線の道路の補間，図 9 では，曲がり角を含む道路の補間，図 9 では，複雑な形状の道路の補間をそれぞれ表している．それぞれの道路図に対応する Google Map の道路を図 11-13 に示す．それぞれの図を比較すると，OpenStreetMap 上には存在しない道路を適切に補間できていることがわかる．以上の実験結果から，提案手法は様々な形状の道路ネットワークに対する補間の有効性が高いことが検証できた．

一方，補間がうまくいかなかった場合も存在する．図 14 は，乱雑な折線を補間した例となる．図 14a は補間結果，図 14b は対象領域の Google map を示す．図 14b が示すように，乱雑な折線を補間した場所には車のサービスステーションがあり，広い駐車場があることが確認できる．道路ネットワークの補間という観点では，不正確な補間となる．一方で，提案手法は駐車場の検出および抜け道となっている駐車場を検出することができることを示している．また，折線の数が狭い範囲で非常に多い場合，道路として補間しないということもできる．



(a) 乱雑な折線補間の結果

(b) 対象領域の Google map

図 14: 乱雑な折線補間の結果と Google map の比較

5.4 補間の精度評価

提案手法の補間の精度は，検出するクラスタの精度に大きく依存する．そのため，DBSCAN のパラメータである $MinPts$ を変化させて精度を評価する．補間の精度を評価するために，補間した道路ネットワークから少数をサンプリングし，Google map と比較することで正しい補間かどうかを検証する．表 3 に $MinPts$ を変化させた場合の生成されるクラスタ数と補間される道路のセグメントの種類の割合を示す(注2)．まず $MinPts$ が 1 の場合，off-road 軌跡が一つでもあればクラスタとして検出されるため，生成されるクラスタ数が多い．また，ひとつの off-road 軌跡から補間される道路は，直線となることが多いため，$MinPts$ が大きい場合と比較して，直線のセグメントが多

(注2)：生成されるクラスタ数と補間される道路は同数となる．

く検出される．$MinPts$ が大きくなるにつれて，検出されるクラスタ数が減少する．特に，$MinPts$ が 3 以上になると，検出されるクラスタ数は少なくなり，直線や折線はほぼ検出されず，複数の折線からなるセグメントの割合が大多数を占めることがわかる．$MinPts$ の値は，データセットのサイズに依存して決める必要があり，今回のデータセットでは，$MinPts$ は，1 もしくは 2 と設定するのが妥当である．

次に，それぞれの $MinPts$ にて生成されるクラスタに対して，10%のサンプリングを実施し，Google map と比較することで補間の精度を検証した．表 4 にそれぞれの $MinPts$ におけるサンプリングしたセグメントの種類とその正解数，および全体での正解率を示す．ここで，正解とは，目視によってその道路が存在するかを確認し，図 14 のような道路以外のセグメントを補間した場合，不正解とした．まず，この表より，提案手法は高い正解率を達成していることがわかる．特に，直線の正解率が高い．これは，直線が一つの道路セグメントから形成されるため，正解となりやすいためである．$MinPts$ が 3 以上になると，多くの場合サービスステーションやガソリンスタンドのような道路以外の補間となるため，複雑な折線の補間が多くなる．そのため，精度が徐々に低下している．

表 3: 各 $MinPts$ に対するセグメントの種類の割合

MinPts	生成されるクラスタ数	直線	折線	複数の折線
1	602	57	346	199
2	207	17	16	174
3	86	4	7	75
4	64	2	0	62
5	53	2	0	51

表 4: 各 $MinPts$ に対する補間の精度

MinPts	サンプリングの数	直線/正解数	折線/正解数	複数の折線/正解数	正解率
1	60	19/17	24/18	17/11	76.6%
2	20	2/1	3/3	15/8	60.0%
3	8	2/0	0/0	6/4	50.0%
4	6	0/0	0/0	6/3	50.0%
5	5	0/0	0/0	5/2	40.0%

6. ま と め

本稿では，不完全な道路ネットワークを用いたマップマッチングと道路ネットワークの補間手法を提案した．不完全な道路ネットワークにマップマッチング手法 [6] を利用して，off-road 軌跡を抜き出し，抜き出したデータを簡単化するし，簡単化された直線または折線の中心点をクラスタリングする．クラスタされた中心点が二つ以上の場合，セグメントを組み合わせて，サブグラフにする．最後，合成されたサブグラフを道路ネットワークに補間する．今後，まず提案手法を実データを用いて評価する．さらに，大規模なデータを用いた場合，実行時間が大きくなることが予想される．そこで，効率的に手法をマップマッチングおよび道路ネットワークの補間を実行する手法を考提案する予定である．

謝　　辞

本研究の一部は，科研費（16H01722, 15K21069）および文部科学省「実社会ビッグデータ利活用のためのデータ統合・解析技術の研究開発」による．

文　　献

[1] Z. Chen, H. T. Shen, and X. Zhou, “Discovering Popular Routes from Trajectories”, In *Proc. ICDE*, pp. 900-911, 2011.

[2] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, and Y. Huang, “Map-Matching for Low-Sampling-Rate GPS Trajectories”, In *Proc. ACM SIGSPATIAL GIS*, pp. 352–361, 2009.

[3] P. Newson and J. Krumm, “Hidden Markov Map Matching Through Noise and Sparseness”, In *Proc. ACM SIGSPATIAL GIS*, pp. 336–343, 2009.

[4] F.C. Pereira, H. Costa, and N.M. Pereira, “An Off-line Map-Matching Algorithm for Incomplete Map Databases”, In . *European Transport Research Review*, 1(3):107–124, 2009.

[5] J.Eisner, S.Funke, A.Herbst, A.Spilner, and S.Storandt, “ Algorithm for Matching and Predicting Trajectories”, In *ALENEX* , pp. 84–95, 2011.

[6] J.-H Haunert and B. Budig, “An Algorithm for Map Matching Given Incomplete Road Data”, In *Proc. ACM SIGSPATIAL GIS*, pp. 510–513, 2012.

[7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”, In *Proc. SIGKDD*, pp. 226–231, 1996.

[8] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun, “Driving with knowledge from the physical world”, In *Proc. SIGKDD* ,pp. 316–324, 2011.

[9] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang, “T-drive: driving directions based on taxi trajectories”, In *Proc. ACM SIGSPATIAL GIS*, pp. 99–108, 2010.