東海大学短期大学紀要第 44 号（2010）

トラッキングシステムにおけるリバースジオコーディングの最適化

八尋　剛規*

（受付　2010 年 9 月 10 日）
（受理　2010 年 12 月 26 日）

An Optimization of the reverse geocoding for the mobile tracking system

by
Takeki YAHIRO*

Abstract

The purpose of this study is to propose a nearest neighbor search method for reverse geocoding on mobile tracking system. This method is based on some characteristic of mobile tracking system and database management system (DBMS).

"Reserve Geocoding" is the process of finding place names, street addresses and other resources for a point location (latitude and longitude). We can find some of ready-to-use reverse geocoding API on the internet. Generally, it searches from the database that includes latitude, longitude, place name and so on, and that database is huge number of records. The place identified by addresses is not a point, but it is a complex polygon. A search method is to calculate whether the polygon contains a target coordinate, and is no simple algorithms and required high-performances computer system.

A proposed method is do search by indexed coordinates (latitude and longitude), next do nearest neighbor search. The database is indexed by latitude and longitude, so it takes to high performance response. By using an indexed database and a proposed method is applicable to shortens the search time, and results of the experiment on the mobile tracking system showed that a method was a practical performance.

Keywords：Reverse geocoding, Nearest neighbor search, GPS, Tracking system

1. 概要

通信技術やモバイル端末の高性能・高機能化及び GPS 機器の普及により位置情報システムが様々な形で普及している。たとえば，GPS ロガーが数千円で購入できるようになり，Google Earth などのソフトウェアと組み合わせて利用すれば，旅の行程を確認することが容易にできる。また，近年モバイル通信環境，特にモバイル時のデータ通信環境が整い，日常的な環境下でも多くの場所でデータ通信が可能となった。これら GPS やモバイル通信環境を利用して，移動ノードの現在位置をリアルタイムで通知するシステムの利用も進んでいる。筆者も GPS 内蔵の携帯電話（以下，GPS 携帯電話と称す）やスマートフォンなどを利用した位置情報共有システムやトラッキングシステムを開発している [1,2]。これは GPS 携帯電話で測位した位置情報をサーバに送信し，地図上にその位置を示すなどの処理を行い，他のユーザと位置情報を共有する Web アプリケーションである。これらのシステムにおいて，位置情報を把握するには地図上にグラフィカルに表示するのが最適であると思われた。しかし，携帯電話など画面表示サイズが物理的に制限される環境では，グラフィカルな地図だけでは，地図のスケール，表示範囲などの兼ね合いから，その地理的位置の認識が困難であることもあった。そこで，グラフィカルな地図に加えて，その位置を文字，すなわち住所で表示することも合わせて行ったところ，その位置認識を行いやすくな

（左） 移動ノードの移動軌跡　（右） 各移動ノードの現在位置

図 1 位置情報共有・トラッキングシステム

*東海大学福岡短期大学
ISSN 0386-8664

った。とくに，土地鑑のある場所ではさらにそれが顕著となる。

これらのシステムで必要になるのが，地図と住所の情報である。地図のベースとなる情報は，国土地理院から数値地図[3]として提供されており，それらの数値情報からグラフィカルな地図を作成することも可能であるし，近年インターネット上のサービスとしてGoogle Mapやその他の地図サービスも提供され，高品質の地図を容易に利用できる環境にある。また住所情報は，街区レベル位置参照情報[4]として国土交通省から提供されている。街区とは，「市町村内の町又は字の名称並びに当該町又は字の区域を道路，鉄道若しくは軌道の線路その他の恒久的な施設又は河川，水路等によって区画した場合におけるその区画された地域」（住居表示に関する法律）のことである。街区レベル位置参照情報はある街区の名称とその代表地点の座標（緯度・経度）の対応表として提供されている。

また，インターネット上のサービスとしてリバースジオコーディングもいくつか提供されている。HTTPのパラメータとして座標を与え，その結果をXML形式で得られるものが大半である。実際，精度・メンテナス・開発時間の点などからこれらのサービスを利用するのが簡単であるが，いずれもWANを介しての利用となり，1座標のリバースジオコーディングの結果を得るのに，0.2～0.4秒ほど要する。筆者が開発しているシステムでは，図1（右）にもあるように，1セッションで複数箇所（多い場合には30箇所以上）のリバースジオコーディングが必要であり，これら既存サービスの利用では，システム全体として十分な応答時間が得られず目的を達しない。

そこで，既存のインターネット上の同サービスを利用せず，独自に街区レベル位置参照情報をデータベース化し，それをSQLで検索し結果を得る手法を選んだ。基本的に，最近傍探索を用いるが，単純な方法では計算量が多くなり実用的ではない。また，筆者が開発している位置情報共有システムでは，その利用特性から道路上，すなわち多くの場合，街区の境界に目的座標がある。さらに海上での利用もあることから，それらも考慮した位置情報の検索手法の開発が必要となった。本稿では，DBMSやトラッキングシステムの特性を活用したリバースジオコーディングの最適化について提案する。

2.　リバースジオコーディング

一昔前までは，GPS搭載機器といえばカーナビゲーションシステム程度しか思い浮かばなかったが，ここ数年で携帯電話やスマートフォン，PC，デジタルカメラなど様々な情報機器に標準搭載されるようになった。GPSの性能も格段に向上している。Assisted GPS（A-GPS）やDifferential GPS（DGPS）などの効果により測位開始までの時間の短縮や測位精度が向上し，日常的な利用がより容易になっている。A-GPSは携帯電話内蔵のGPS，DGPSはカーナビゲーションシステムや船舶のGPSで利用されている。

GPSを用いた位置測位の結果は緯度・経度により提供され，位置が特定される。緯度・経度は数値であり，地図上で計算機的に位置を特定するには有効であるが，我々人間にとっては緯度・経度から，それが実際その場所であるのかを判断するのは困難である。人間が，ある場所を特定する材料は一般的に住所や地名といわれる文字の並びであり，実際，公文書でも住所を用いることが多い。

この緯度・経度から該当する住所を特定することをReverse Geocoding（リバースジオコーディング，逆ジオコーディング）という。逆に住所から緯度・経度を特定することをGeocoding（ジオコーディング）という。

ジオコーディングやリバースジオコーディングを実現するには，緯度・経度の座標情報と住所の対応表が必要となる。ここで注意すべきことは，ある住所で示される場所は1箇所の「点」ではなく「複雑な多角形」である。したがって，1つの街区に対して，複数個の頂点座標情報も必要となり，そのデータ数は膨大なものとなる。日本国内の場合，○○市○○N丁目M（例えば，福岡県宗像市田久1丁目9）の街区レベルでは，11,240,217箇所となる。これに各頂点の座標を加えると，さらに数十倍近いデータ数となる。この時，ある座標(X,Y)の住所を検索するには，その座標が，どの街区の多角形に内包されるかを調査すればよい。単純なアルゴリズムであるが，その分そのデータ数に比例して計算量が多くなる。それ以前に，この面情報を入手するのが困難であるという現実問題もある。現状，無料で利用できる同種のデータは存在しない。

我々が入手可能なデータの一つに国土交通省が提供している街区レベル位置参照情報がある。これは街区単位の位置座標（その街区の緯度・経度）データである。これはその街区の代表点で示される「点」のデータであるため，ある座標に対してリバースジオコーディングを行おうとしても，それに最も近い街区の代表点を検索し，近似的に結果を求めることとしかできない。これらの解法として最近傍探索（Nearest Neighbor Search, NNS）が知られている。

最も単純な解法として線形探索がある。これは，座標(Y1,X1)，(Y2,X2)の距離を求める関数をdist(Y1,X1, Y2,X2)[*1]と定義すれば理論上select * from gis order by dist(latitude, longitude, Y, X) limit 1の単純なSQLで求めることができる。しかし，この方法では，データベースのすべてのレコードに対して距離の計算を行わなければならず，レコード数が少ない場合は問題ないが，レコード数が多いと，それだけ膨大な計算量となり実用的ではない。実際この手の解法としては，ボロノイ図（Voronoi Diagram），あるいは離散ボロノイ図（Discrete Voronoi Diagram）やバゲット（Bucket）法を用いて高速に検索さ

れることが多く，計算量は log(N)程度に抑えられる。ボロノイ図は,ある平面空間の任意の複数個の点に対して,どの点にもっとも近いかにより領域分けされた図であり，各境界線は各点の二等分線となる。その境界線をボロノイ境界という（ボロノイ図については文献 [5]が詳しい)。しかし，実際の街区の境界とボロノイ境界が一致するとは限らず，これも近似的な結果を得ることになる。この離散ボロノイ図を作成しデータベース化しておけば，事前に座標(Y,X)の丸め処理を行い，select * from gis_voronoi where latitude = Y and longitude = X で処理可能となる。しかし，離散ボロノイ図を作成し，その結果を保存しておく必要もあり，その時間とデータ量は膨大なものなる。近年，頻繁に行われている市町村合併が実施される度にボロノイ図の再計算が必要となることも考えられる。

このほか，街区レベル位置参照情報を一定エリア毎にクラスタリングし，そのクラスターで一次検索を行うことにより処理の高速化を行う手法も提案されている [6]。また，国土交通省が提供している国土数値情報（数値地図）もメッシュコードによりクラスタリングされ，検索の高速化の一要因となっている。しかし，対象とするエリアがひとつのクラスターに限定されず，隣接するクラスターも検索する必要がでてくることや，クラスター毎のデータ数も都市部と村落部では違いがあることなどから，そのクラスタリングも簡単ではない。

3. トラッキングシステムにおけるリバースジオコーディング

<3.1> 一般的なリバースジオコーディング

一般的に利用されるリバースジオコーディングの座標としては，陸上のあるオブジェクト（建物など）であることが多い。したがって，先に述べたように内包する点を検索するアルゴリズムで目的を達成できる。

<3.2> トラッキングシステムにおけるリバースジオコーディング

筆者が開発している位置情報共有システムは，移動体（以下，移動ノード）のトラッキングシステムとして利用もされており，道路上などを移動中に位置情報が発信され，それに対するリバースジオコーディングの必要性が高い。またトラッキングシステムの特徴のひとつとして，移動軌跡を正確に残すことにある。このため，移動ノードが変針した時（交差点を曲がったなど）に位置情報が登録される。このように陸上では，道路上や交差点におけるリバースジオコーディングが必要となる。国内では，道路を街区の境界と設定することが多く，したがって，交差点などは 2 つ以上の街区の境界であることも多い。よって，GPS の測位誤差などが含まれることを考慮すれば，詳細にリバースジオコーディングを行う意味合いが小さくなる。すなわち，各街区を構成する多角形の頂点座標のデータを利用して厳密なリバースジオコーディングを行う必要はなく，最近傍探索の手法を用いて行うことで十分にその目的を達成できる。

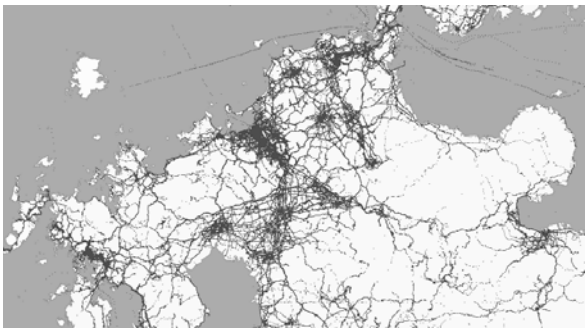

図 2 トラッキングシステムの軌跡

また，陸上のトラッキングシステムにおいて，リバースジオコーディングが必要となる地点は，実際の陸上部の面積の極わずかである。図 2 は九州地区におけるトラッキングシステムの表示例で，移動ノードから発せされた位置情報を地図上にプロットしたものである。この図からもわかるように，陸上部においては，道路沿いを中心に軌跡が残っており,陸上部における面積比は小さい。このことから，陸上全域に対して離散ボロノイ図を事前に作成する必要性も小さい。

一方，船舶におけるトラッキングシステムである自動船舶識別装置（Automatic Identification System , AIS）は，当然ながら海上から位置情報が発信される。海上の場合，地図上で現在位置を示すには陸上の場合と何ら変わりなく処理できるが，文字すなわち住所で表示することは難しく，一般的に「○○の沖○○Km」のように表現される。この場合，前述したリバースジオコーディングの手法では処理できず,「陸上の最近傍点の住所」を検索せざるを得ない。

図 3 AIS（船舶など海上移動ノード）における表示例

左端は進行方向と速度，右は最近傍の住所およびその地点からの方位と距離

一部のカーナビゲーションや Google Map でも実現されているが，リバースジオコーディングの情報に道路名を付加している場合がある。この道路名は陸上におけるトラッキングシステムにおいて非常に効果的な情報である。さらに交差点名称がわかれば,「国道○号線○○市○○交差点付近」という表記がより効果的であり，実際，

ラジオの交通情報番組などでもそのフレーズをよく聴く。米国などでは全ての道路に○○ストリートなどの名称がつけられているが，国内では国道○○号線，県道○○号線の名称がつけられているほか，一部の道路には○○通り，○○△△線のように別途名称がつけられていることがあるが，米国のそれに比較すれば，その情報量はわずかであり，加えて，その入手可能なデジタル情報も少なく，今回は対象外とした。

<3.3>　トラッキングシステムにおける住所表示の必要性

トラッキングシステムでは，グラフィカルに地図上に現在位置や軌跡を表示できれば，それで十分だと予想され，筆者が開発している同システムでも当初は地図上にこれらを描画するのみであった。しかし，ケータイなど物理的な画面サイズや解像度，ユーザインタフェースが限られる環境では，それで十分とはいえないことが分かってきた。実際，同システムにおいて移動ノードの軌跡を地図表示と住所表示で行えるようにしてユーザの利用状況を調査したところ，表 1 のような結果となった。思いのほか, 住所表示による位置参照を行うユーザが多い。聞き取り調査を行ったところ,「土地鑑のある場所では住所表示で大抵の位置関係が把握できる」, 逆に「土地鑑のない場所での地図表示では地図のスケールによってはそこがどこであるのかの判断がつかず，住所表示を頼る」などの回答を得た。一般的に地図上にも代表的な住所表示は行われるものの，それでは不十分な場合もあり，文字列による住所表示を併用することで位置情報を把握が的確に行われているものと推測される。

表 1 地図と住所の利用割合

適用	割合
地図のみ	54%
移動軌跡を住所で表示	26%
周辺ノードの位置を住所で表示	19%

<3.4>　同時に多数のリバースコーディングの必要性

複数の移動ノードの現在位置や移動履歴を地図表示にグラフィカルに表示するだけでなく，それを住所表示で行う必要があり, 同時に 30 以上のリバースジオコーディングを要求する。今日，Web は 2 秒ルール [7,8]といわれるような時代であり，これから単純計算すれば，1 つのリバースジオコーディングに許される時間は，0.06 秒以内である。ネットワークやその他の処理のオーバーヘッドを考慮すれば, さらに 1 秒に 30 件程度処理できる処理速度が要求される。

<3.5>　トラッキングシステムに求められるリバースジオコーディングの性能

したがって，表 2 に示す性能を持つリバースジオコーディングシステムが必要となった。

Google や Yahoo その他で提供されているリバースジオコーディング API は，表 2 に示す必要性能の内，精度は十分満たすものの，速度・回数の点では不安が残る。このため，ローカルシステムとして利用できるリバースジオコーディングを開発することとした。

表 2 トラッキングシステムが必要とするリバースジオコーディングの性能

項目	要件
精度	大字・町丁目レベル　「付近」程度 海上においては最近傍の住所
速度	30 件/秒　以上
回数	制限なし

4. 提案手法

<4.1>　システム構成

システムはハードウェアとして Xeon 2.33GHz, RAM 2GB, HDD 135MB/s (RAID 1), OS として, FreeBSD 7.2, データベースマネージメントシステム（RDBMS）として PostgreSQL 8.3.7 を利用した。この RDBMS 上に街区レベル位置参照情報を表 3 に示すテーブル（テーブル名 gis）として保存した。総レコード数は 11,240,217 である。

表 3　テーブル gis の構造

Column	Type	適用
Pref	character varying(20)	都道府県名
City	character varying(20)	市区町村名
Town	character varying(20)	大字・町丁目
Address	character varying(20)	街区符号・地番
Latitude	double precision	緯度
Longitude	double precision	経度

Indexes: "gis_idx_pos" btree (latitude, longitude)

<4.2>　座標（緯度・経度）インデックスによる高速化

テーブル gis に対し，緯度(latitude)及び経度(longitude)でインデックスを作成している。レコード数の多いデータベースを検索する上でインデックスは非常に有効であることは明示であり，例えば，ある緯度・経度のレコードを検索する場合のコストは，インデックスなしでは 0.00..348256.26 であるのに対し，インデックスありでは 0.00..10.00 となる。実際の検索速度も 22993.338 ms に対し，0.037 ms と高速化されインデックス化の効果が表れている。

EXPLAIN ANALYZE select * from gis where latitude = 33.80 and longitude = 130.565;

インデックスありの場合

Index Scan using gis_idx_pos on gis (cost=0.00..10.00 rows=1 width=72) (actual time=0.014..0.014 rows=0 loops=1)

Total runtime: 0.037 ms

インデックスなしの場合

Seq Scan on gis (cost=0.00..348256.26 rows=1 width=72) (actual time=22993.307..22993.307 rows=0 loops=1)

Total runtime: 22993.338 ms

※EXPLAIN の結果から抜粋

しかしながら，目的とする座標（たとえば，北緯 33.800 度，東経 130.565 度:東海大学福岡短期大学/福岡県宗像市付近の座標）を検索する場合，select * from gis where latitude = 33.800 and longitude = 130.565 では結果を得ることはできず，これに最近接する座標のレコードを検索する必要がある。この場合，理論的には select * from gis order by dist(latitude, longitude, 33.800, 130.565) limit 1;となるが，座標のインデックスが利用されないため膨大な計算量となり実用的でない。実際，コストは 3158310.50..3158310.51 であり，時間も 145115.000 ms という結果となる。

```
EXPLAIN ANALYZE select * from gis order by dist( latitude, longitude, 30.80, 130.565 ) limit 1;
  Limit  (cost=3158310.50..3158310.51 rows=1 width=72) (actual time=145114.959..145114.961 rows=1 loops=1)
  Total runtime: 145115.000 ms
```

<4.3> 検索レンジによる座標絞り込みとインデックスの効果的な利用

そこでインデックスされた座標を効率的に利用するために，まず座標で絞り込みを行い，最近接する座標をもつレコードを探す。次に示す例は，絞り込みを行う座標の範囲（以下，検索レンジ）を 1 文の SQL で行った結果である。この絞り込みで得られるレコード数は 428,995 となり，このレコードすべてに対して距離の計算を行うため最終的な計算量は依然多くなる。

```
EXPLAIN ANALYZE select * from gis where 33 < latitude and latitude < 34 and 130 < longitude and longitude < 131 order by dist( 33.800, 130.565, latitude , longitude ) limit 1;
  Limit  (cost=66600.43..66600.43 rows=1 width=72) (actual time=5607.796..5607.798 rows=1 loops=1)
  Total runtime: 5607.848 ms
```

すなわち，座標による絞り込みを適切に行い，その結果のレコード数を減らせばその後の計算量が少なくなり，高速に検索することが可能となる。例えば，座標の検索レンジを 30 分（面積比 25%）にすれば， 対象となるレコードが 131,581 件に絞られ，時間が短縮される。

```
EXPLAIN ANALYZE select * from gis where 33.5 < latitude and latitude < 34 and 130.5 < longitude and longitude
< 131 order by dist( 33.800, 130.565, latitude , longitude ) limit 1;
  Limit (cost=35886.53..35886.53 rows=1 width=72) (actual time=1851.606..1851.608 rows=1 loops=1)
  Total runtime: 1851.653 ms
```

実際，座標の検索レンジを 0.1 度にまで狭めると，検索時間は 80ms まで短縮でき十分実用的なレベルに達する。

```
EXPLAIN ANALYZE select * from gis where 33.8 < latitude and latitude < 33.9 and 130.5 < longitude and longitude < 130.6 order by dist( 33.800, 130.565, latitude , longitude ) limit 1;
  Limit  (cost=4182.16..4182.16 rows=1 width=72) (actual time=80.524..80.526 rows=1 loops=1)
  Total runtime: 80.569 ms
```

このように，座標の検索レンジを小さくすれば，その後の距離計算数が少なくなり，結果として検索時間が短くなる。しかし，範囲を小さくしすぎると，その範囲内に街区の代表点が存在しない可能性もでてくる。実際，都市部では一つの街区の範囲が小さく，逆に村落部のそれは大きいことが一般的で，座標の検索レンジを一意に定めることが難しい。

<4.4> 検索アルゴリズムと検索レンジ初期値

そこで，インデックス化された座標を一次検索として利用することにより高速化を図ることとした。そのアルゴリズムを次に示す。

手順 1) 検索する座標(緯度,経度)を($lat, $lng)とする
手順 2) 検索範囲 $r の初期値を定める
手順 3) $y1 = $lat - $r; $x1 = $lng - $r;
　　$y2 = $lat + $r; $x2 = $lng + $r;
手順 4) select * from gis
　　where $y1 < latitude and latitude < $y2
　　and $x1 < longitude and longitude < $x2
　　order by dist(latitude, longitude, $lat, $lng)
　　limit 1;
手順 5) 手順 4 の結果，該当するレコードが存在しない場合
　　$r = $r * 2;
　　手順 3) に戻る

検索レンジの最適初期値を実験的に求めるために，検索レンジの初期値を 0.00003125 から倍数的に変化させながら，都市部，村落部，海上のある地点を中心として，緯度・経度それぞれ 0.001 度の範囲でランダムに 20 か所生成した座標に対してリバースジオコーディングを行い，その所要時間を計測した。なお，所要時間には実際にリバースジオコーディングで街区情報を得る時間の他に，検索結果のキャッシュへの書き込み，ログ情報の記録などの付随する処理も含んだ時間である。中心とした場所は，福岡県宗像市（東海大学福岡短期大学付近），福岡市博多区（博多駅付近），東京都渋谷区（東海大学代々木校舎付近），北海道川上地方　美瑛，北海道川上地方音威子府村，及び宗像市沖の海上 4km 地点とした。その結果を図 4 に示す。

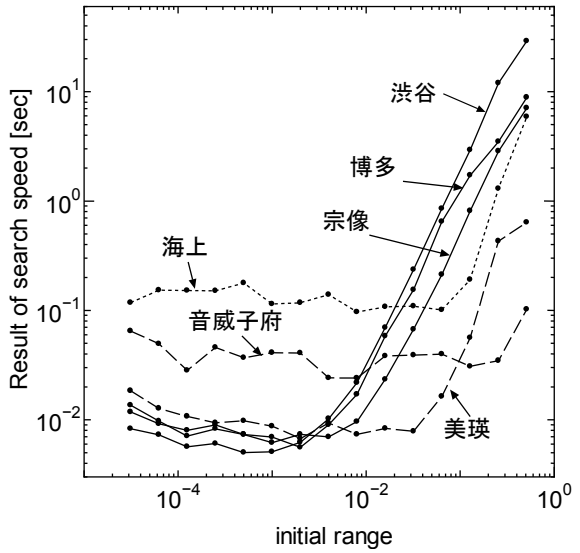

図 4 都市部と村落部における
検索レンジ初期値と検索速度の関係

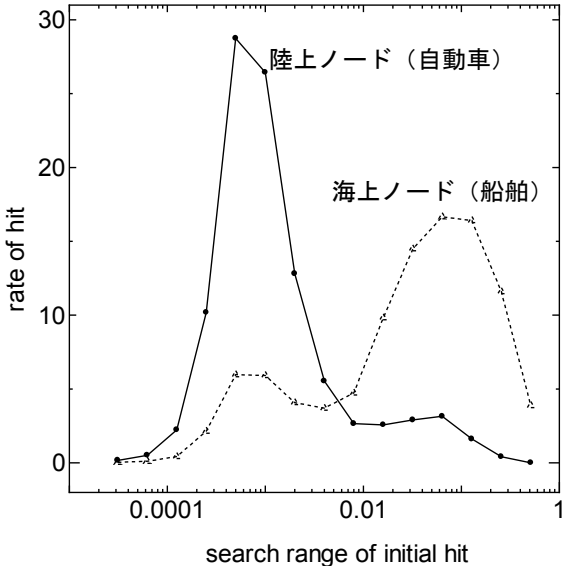

図 5　陸上と海上における検索レンジの相違

都市部（宗像，博多，渋谷）では，検索レンジ 0.0005～0.004 付近で最速となるが，これを超えると指数的に処理速度が遅くなる。これは都市部では一定範囲内に存在する街区の代表地点が多数存在し，そのすべてに対し距離計算が生じることに起因する。一方，村落部（北海道音威子府）では，検索レンジ全般を通して処理速度に変化が見られない。検索レンジが小さいときは，その検索レンジ内で街区の代表点が存在せず，検索レンジを 1 レベル広げて再検索を行うためである。海上でも同じような理由により，村落部と同様の結果となる。

表 4 検索レンジと検索成功数（陸上ノード）

検索レンジ	検索成功数	累積割合
0.00003125	3454	0.17%
0.0000625	10379	0.69%
0.000125	44753	2.93%
0.00025	203431	13.12%
0.0005	574108	41.87%
0.001	528134	68.31%
0.002	255855	81.12%
0.004	110690	86.66%
0.008	53064	89.32%
0.016	51446	91.90%
0.032	58041	94.80%
0.064	62972	97.96%
0.128	32288	99.57%
0.256	8293	99.99%
0.512	218	100.00%

5. 実証評価

<5.1>　実証環境と評価

実験的に検索レンジの初期値を 0.00003125（約 6m 四方）と設定し，トラッキングシステムのリバースジオコーディングシステムを運用した。そして，どの検索レンジにおいて住所の検索に成功したかを調査した結果を表 4 及び図 5 に示す。その結果，自動車による陸上移動ノードの場合，検索レンジは 0.0005 の時に検索成功数が約半数に達することから，検索レンジの初期値を 0.0005 とした。実際の距離にして約 100m 四方となる。

一方，船舶における海上移動ノードの場合，検索レンジに 2 つのピークが見られる。一つは自動車の場合と同じく検索レンジ 0.0005 で小さなピークがあり，他に 0.064（約 12km 四方）で最大のピークが見られる。0.0005 のピークは港に停泊中の船舶から出された位置情報に対するリバースジオコーディングであると予想される。

<5.2>　**地域による検索レンジの差**

国内の陸上移動ノード（主に自動車）から発せされた位置情報に対してリバースジオコーディングを行った結果を次に示す。検索レンジが小さい値で検索された場所は，一つの街区エリアが小さい都市部が必然的に多くなり，逆に検索レンジが大きい値で検索された場所は，村落部が多くなっている。

検索レンジが最小値で結果が得られた住所の例

- 埼玉県さいたま市桜区新開二丁目
- 東京都墨田区両国一丁目
- 北海道札幌市中央区宮の森二条十六丁目
- 茨城県水戸市松本町
- 愛媛県松山市保免西二丁目
- 東京都江戸川区一之江七丁目
- 北海道札幌市中央区南一条西二十五丁目

- 北海道札幌市豊平区豊平五条十三丁目
- 愛知県名古屋市東区東桜二丁目
- 福岡県北九州市小倉南区下城野二丁目

検索レンジが最大値で結果が得られた住所の例

- 熊本県天草市牛深町
- 北海道標津郡中標津町字俵橋
- 北海道夕張市登川
- 三重県熊野市須野町
- 東京都三宅村坪田
- 北海道根室市穂香
- 北海道斜里郡斜里町字以久科北
- 北海道根室市穂香
- 北海道苫前郡羽幌町字築別
- 北海道釧路市桜田十三線

<5.3> 検索結果のキャッシュと有効性

リバースジオコーディングの結果は，データベース上の別テーブル（テーブル名 giscache）にキャッシュ情報として記録する。システムが実際にリバースジオコーディングを行うときは，このキャッシュに対して検索を行い，それでも見つからない場合は，本来の街区レベル位置参照情報テーブルから検索を行う。

トラッキングシステムとしては移動ノードだけでなく固定ノードの位置情報も含まれるため，ある座標に対して複数回のリバースジオコーディングが要求される場合も多い。過去 21 ヶ月間（2009 年 12 月～2010 年 08 月）の統計で，同一キャッシュに対して最大 95,320 回のヒットが確認されている。移動体ノードでも，最大 29,003 回で，平均 10.6 回と効率的にキャッシュが機能している。

<5.4> 他リバースジオコーディングサービスとの比較

HTTP 経由の API として提供されているリバースジオコーディングとの速度比較を行った。実際に筆者が運用を行っているトラキングシステムに記録されている移動ノードの最新座標 50 箇所について，提案手法によるリバースジオコーディングシステムとその API(HTTP)経由及び他システムで検証を行った。

1. 提案システム（PHP からデータベースサーバへ直接接続）
2. 提案システムを API(HTTP)でアクセス
3. Google Geocoding API [9)]
4. Yahoo リバースジオコーダー [10)]
5. 簡易逆ジオコーディングサービス（独立行政法人農業・食品産業技術総合研究機構）[11)]
6. 逆ジオコーディング API（NPO 法人　情報活用センター）[12)]

上記 API は，それぞれ対象となる地域の違い，ベースとなる街区データの収録状況，サーバの負荷，システム性能などに差があるために，客観的に正確な比較は不可能である。また，他 API（上記リストの 3～6）は HTTP 経由でのアクセスになるため，ネットワークや Web サーバ，その他のオーバーヘッドを考慮する必要がある。検証時点でのネットワーク遅延は 40ms～70ms であった。比較のため，他 API 同様に，本提案手法によるシステムも HTTP 経由の API でアクセスし，XML 形式で結果を返すようにして(上記リストの２)，できる限り他 API と同一条件になるようにして検証を行った。この API を設定した Web サーバへのネットワーク遅延は 0.25ms 程度である。これら API 等へのアクセスは，1 箇所のリバースジオコーディングの結果が得られたら，できる限り間をあけず次のアクセスを行った。HTTP 経由によるアクセスはリクエスト毎に TCP セッションを張りなおしている。その結果を図 6 に示す。

トラッキングシステム内にリバースジオコーディングシステムを内包した場合（図 6 の Original(Direct)）は 1 リクエスト当たり平均 0.004 秒で処理できている。これを他のサービス同様に HTTP 経由の API（図 6 の

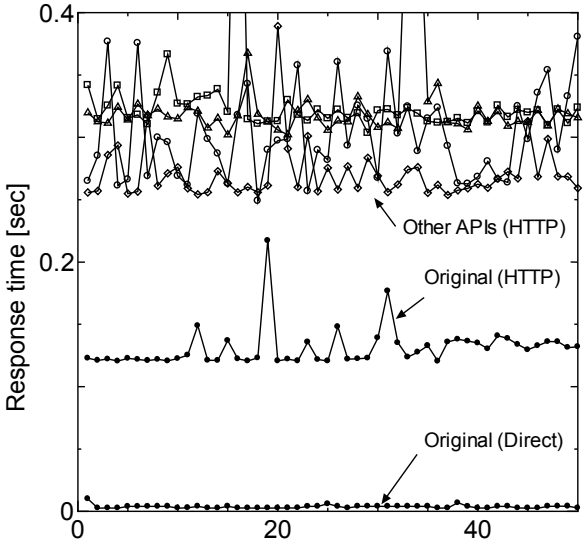

図 6 他サービスとの比較

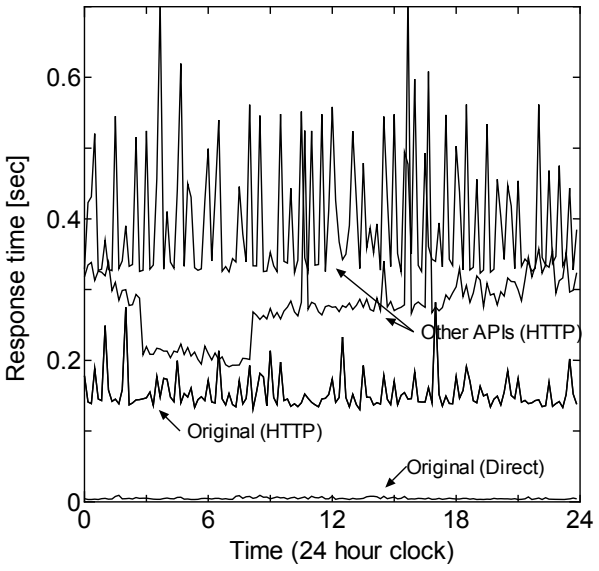

図 7 時間帯別応用時間の推移

Original(HTTP)）で検証したところ，平均 0.12 秒となり，HTTP 経由によるネットワークや Web サーバソフトウェアによるオーバーヘッドが大きいことがわかる。他システムにおける検証では，途中のネットワークの輻輳やサーバの負荷などの影響と思われる処理速度の揺らぎがあるが，平均 0.25 秒であった。また，それぞれのサービスに対し 10 分間隔で，ランダムに発生させた座標 10 箇所のリバースジオコーディングを行う，その平均応答時間を計測した結果の一部を図 7 に示す。時間帯により 0.2 秒～0.5 秒程度まで変化がある。図 6 はこの平均的な結果が得られた時のものである。

<5.5>　本手法の問題点

一つの街区が大きく，その街区が細長い場合や街区の代表点がその街区の中心でない場合，実際とは異なる街区を検索してしまうことがある。これを解決するには，実際の街区の空間的中心を座標としてデータベース化する，あるいは街区を多角形としてデータベース化するしかない。

6. まとめ

本稿では，DBMS のインデックスを利用した検索高速化及びトラッキングシステムにおける位置情報の特徴を利用したリバースジオコーディングの手法を提案した。提案した手法は，トラッキングシステムの特徴として高精度なリバースジオコーディングは要求されないため，近似的な結果が得られる最近傍探索を用いるものである。また，高速化を図るため，クラスタリングによる一次検索のかわりに，インデックス化された座標を用いて一次検索を行い，最近傍の計算の対象となるデータを限定することとした。この際，一次検索の座標範囲は，0.0005 度が最適であることをシミュレーションにより示した。また，提案手法をトラッキングシステムに内包した場合，必要な処理能力を得られることを示した。

今後の課題として，更なる高速化を図りたい。このため，検索レンジ初期値の最適化を地域別（都市部と村落部），あるいは陸上・海上別に分けて設定することなどを検討している。たとえば，大雑把に都市部・村落部や陸上・海上により区分した離散ボロノイ図を併用する。さらに，システム開発当初に比較してメモリーも潤沢に搭載できるようになり，メモリー使用量考慮型から HDD アクセス頻度考慮型へプログラムを改良することも考えられるであろう。また異なる街区を検索してしまう件については，市町村レベルで街区情報を多角形でデータベース化し内包点を探索して市町村を特定し，そのレベル以下を本手法で検索するなどの手法をもちいて問題解決を行いたい。これらにより更なる性能向上を目指したい。

引用文献

1) 八尋剛規: 「KetaiTracker -携帯電話で APRS!!-」,CQ Ham Radio, No.768, 2010.06 号， pp.96-99, CQ 出版社, 2010
2) 八尋剛規,小川兼司: 「GPS ケータイを利用した位置情報共有システムの開発」, モバイル'10 研究論文集, pp.17-22 ,モバイル学会, 2010
3) 国土交通省国土計画局:国土数値情報ダウンロードサービス, http://nlftp.mlit.go.jp/ksj/
4) 国土交通省国土計画局:位置参照情報ダウンロードサービス, http://nlftp.mlit.go.jp/isj/
5) 杉原厚吉: 「なわばりの数理モデル」, 共立出版, 2009
6) 松田晃太郎: 「地理情報のクラスタリングによる逆ジオコーディング処理の効率化」, 秋田職業能力開発短期大学校紀要　第 13 号,2008
7) 「Zona Market Bulletin」, Zona Research , 2001
8) 「eCommerce Web Site Performance Today」, Forrester Research, Inc, 2009
9) http://code.google.com/intl/ja/apis/maps/documentation/javascript/v2/services.html#ReverseGeocoding
10) http://developer.yahoo.co.jp/webapi/map/openlocalplatform/v1/reversegeocoder.html
11) http://www.finds.jp/ws/rgeocode.php
12) http://geocode.didit.jp/

*1 最近傍探索を行う場合，距離を求める関数 dist は，実際の距離計算を行う必要はなく簡易的に距離の比較ができればよいが，今回はトラッキングシステムの機能上，実際の距離を算出する必要があるため，距離計算を行っている。