平成 29 年度　特別研究報告書

ＧＰＳを用いた市バスの位置情報取得アプリの研究

龍谷大学　理工学部　情報メディア学科

T120508　学生氏名　山崎和久

指導教員　三好　力　教授

内容梗概

IoTの発達で、交通機関は大きな発展を見せている。電車やバスなどの発着時刻の確認や遅延情報など今では、携帯端末や電子時刻表などで確認できます。しかし、交通機関の遅延は必ず発生する。特に、バスは渋滞や乗客の乗り降りで他の交通機関に比べて遅延する要因が多い。最近では京都市内のバス停で到着案内システムが導入されているが、これは今バスがど このバス停を出発したことしか分からない。昨今スマートフォンなどの携帯端末が爆発的に普及している中で、携帯端末だけを用いて問題を解決することは世界中で見受けられる。そこから公共交通機関はバス到着案内システムのような各バス停に設置しなければならない大掛かりなものを作るより携帯端末のアプリケーション内で問題を解決できればコストも削減できる上に、コンパクトなシステムになると考えられる。本研究では、アンドロイドのGPSセンサーを用いて不特定多数の乗客のアンドロイド端末からバスの位置情報を 取得し、バスの動きを視覚化するアプリを提案し、その性能を検討する。

目次

第1章　緒論

1.1　研究背景

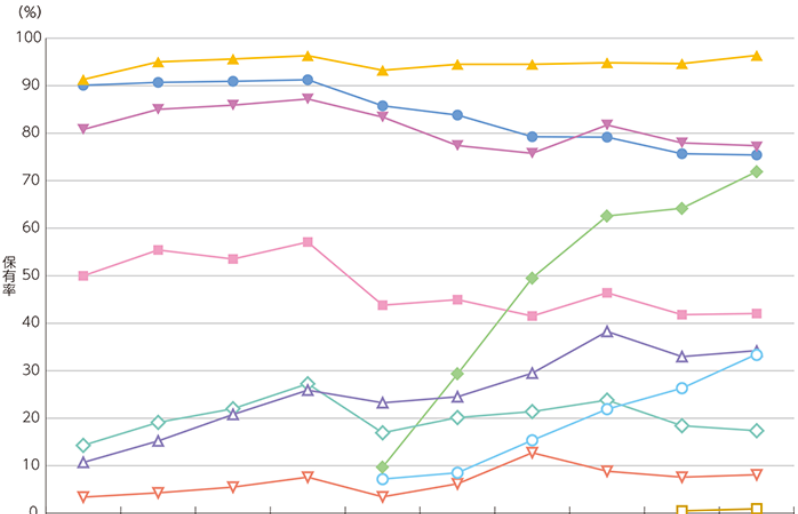
近年、IT 技術の進歩によりスマートフォンのアプリを用いたサービスが普及しており、スマートフォンは人々の生活にとってなくてはならないものになってきた。総務省の調べでは、2015年末の時点でスマートフォンの普及率は72.0％である。[1]その普及率を図1.1に示す。2017年現在では、10代、20代のスマートフォン普及率が90％を超えている。[2]その普及率を図1.2に示す。

その中で、公共交通機関もスマートフォンの普及により影響を受けている。今では2.4に後述する通り、公共交通機関の時刻表や遅延情報をスマートフォンからインターネットを利用し、情報を入手することができる。本来、駅やバス停で得る情報をインターネットに接続されたスマートフォンが片手にあればどこにいても公共交通機関の情報を取得することができるようになった。

しかし、便利になったとはいえ問題点もある。ここで、バスについて考えてみる。バスは電車と違い、遅延する要因が多い。例えば、バスは一般道路を走るので、一般車と同じ道を走ることになる。そうすると、渋滞に巻き込まれる可能性や道が混む時間帯が存在するという問題が発生する。そこで、2.3で後述するスマートフォンなどのアプリケーションを用いて、遅延情報を得ることはできるが、バス停でバスをまっている人は、不安を感じてしまう。京都市内では、2.2で後述するバスの到着案内システムが導入されているがこれは、バスがどこにいるのかが分かるのではなく、あくまで、どのバス停にバスが到着したかが分かるだけなので遅延が起こった場合、待っている人はバス停間のバスの動きがわからない上に、全てのバス停に設置されているわけではない。

本研究では、アンドロイド端末の GPS センサーを用いて、バス内の不特定多数の乗客の位置情報と移動速度を取得し、サーバーでその位置情報と速度を元に値の近いものをひとつのグループをバスとしてまとめ、アンドロイド内のアプリにバスの位置情報を返しスマートフォンの画面上に位置情報を表示するアプリの開発を目標とする。

これにより、スマートフォン上でマップを用いてバスの正確な位置情報と動きを視覚化することで、遅延で不安を感じている人が安心を得ることが期待される。



	2006 (n=4,999)	2007 (n=3,640)	2008 (n=4,515)	2009 (n=4,547)	2010 (n=22,271)	2011 (n=16,530)	2012 (n=20,418)	2013 (n=15,599)	2014 (n=16,529)	2015(年) (n=14,765)
固定電話	90.1	90.7	90.9	91.2	85.8	83.8	79.3	79.2	75.7	75.6
FAX	50.0	55.4	53.5	57.1	43.8	45.0	41.5	46.4	41.8	42.0
携帯電話・PHS	91.3	95.0	95.6	96.3	93.2	94.5	94.5	94.8	94.6	95.8
スマートフォン	–	–	–	–	9.7	29.3	49.5	62.6	64.2	72.0
パソコン	80.8	85.0	85.9	87.2	83.4	77.4	75.8	81.7	78.0	76.8
タブレット型端末	–	–	–	–	7.2	8.5	15.3	21.9	26.3	33.3
ウェアラブル端末	–	–	–	–	–	–	–	–	0.5	0.9
インターネットに接続できる家庭用テレビゲーム機	10.7	15.2	20.8	25.9	23.3	24.5	29.5	38.3	33.0	33.7
インターネットに接続できる携帯型音楽プレイヤー	14.3	19.1	22.0	27.3	17.0	20.1	21.4	23.8	18.4	17.3
その他インターネットに接続できる家電(情報家電)等	3.4	4.3	5.5	7.6	3.5	6.2	12.7	8.8	7.6	8.1

図1.1　情報通信端末の世帯保有率の推移

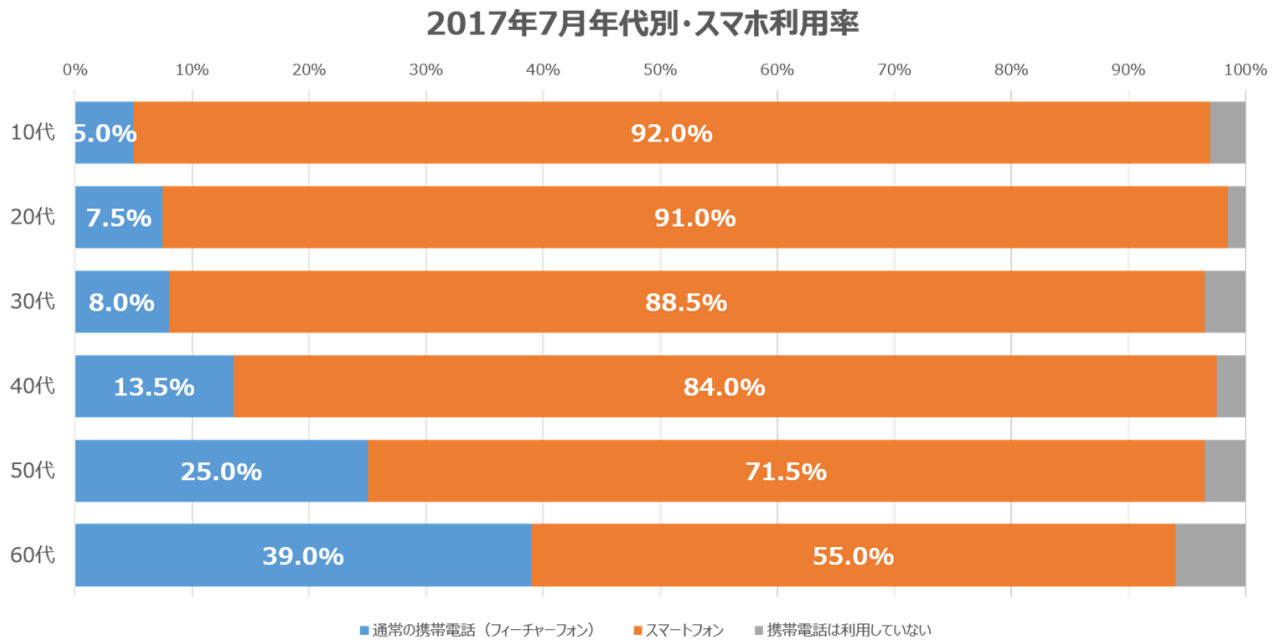


図1.2　2017年7月 年代別・スマホ利用率

第2章　既存技術と問題点

2.1　Beacon

2.1.1　Beacon とは？

Bluetooth という信号を発信する発信機で、信号を数秒に一回、半径数十メートルに発信し、スマートフォンなどの受信端末が、その範囲内に入ってくると信号を受け取ったスマートフォンなどの情報端末は情報をサーバーに自動的に送る。サーバー側は、スマートフォンが Beacon からの信号を受け取ったという情報を受けとる。この技術を応用したのが位置情報サービスで、これを複数の発信機で行うと、スマートフォンの持ち主がどこからどこへ移動したかが分かる仕組みである。だが位置情報を送るだけではなく、その場所にいるスマートフォンユーザーに何らかのメッセージを送りたい場合、事前にサーバーにこのエリアに入ったユーザーにメッセージを送るという指令を事前に与えておくことが可能で、こうすることでこの範囲に入ったユーザーに一斉にメッセージを送ることが可能になる。これを応用したものがクーポンなどのプッシュ通知である[3]。これを図2.1に示す。

図2.1　Beacon とプッシュ通知のイメージ図

2.1.2 Beacon の種類

・雪崩ビーコン

登山やスキーを行う際に、雪崩に遭遇する可能性のある場所で用いられる小型のビーコンで、雪に下敷きになってしまった人の位置情報をビーコンの電波から探索することができる。これを図2.2に示す。

図2.2　雪崩ビーコンのイメージ図

・VICS（vehicle information and communication system）

道路交通情報通信システム。道路脇の発信機を経由して、VICS センターから、自分の車の位置、渋滞、事故などの情報を専用の受信機を利用し自分のカーナビの画面に表示する情報システムである。これを図2.3、図2.4に示す。

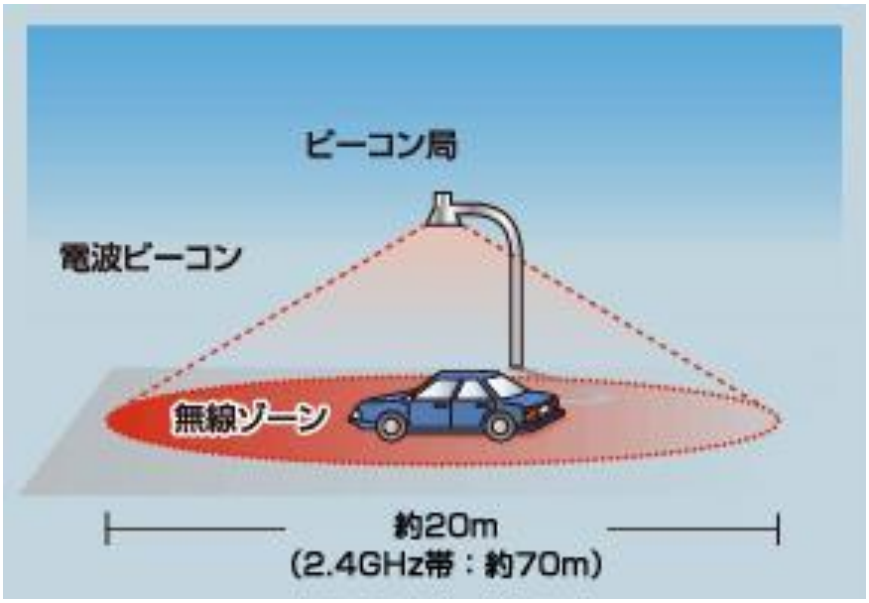


図2.3　VICS イメージ図

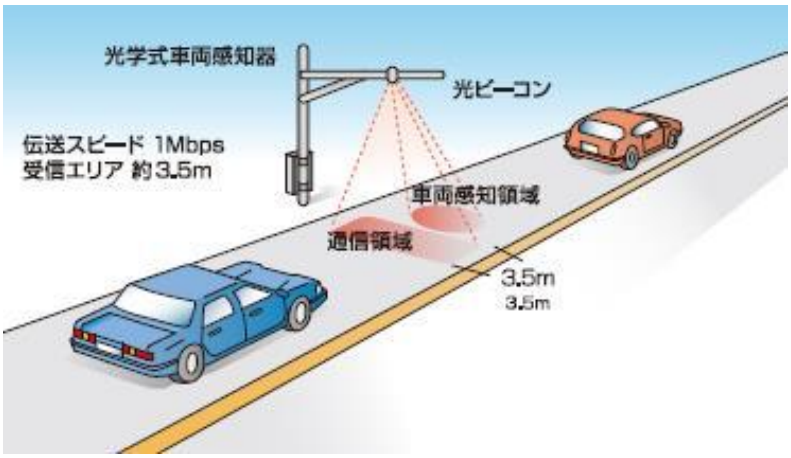


図2.4　VICS イメージ図2

2.2　バス停到着案内システム

バスの運転手の次のバス停アナウンスのボタン操作と連動し、バスがどのバス停へ向かっているか識別する。その情報が専用無線経由で中央の管理センターに送られそこから各バス停へ情報を送る。送られた情報を元に、バスの表示位置を変更する。IoT の発展によりバス停もインターネットに接続されるようになった[4]。これを図2.5に示す。

図2.5　バス到着案内システム

2.3 バス NAVITIME

バス停を TOP 画面に設定すると、次の発車時刻がすぐにわかり、時刻表のカウントダウンやバスの接近情報がわかる。自分の現在地から近い周辺のバス停も検索することができる。乗り換え案内も調べることができ、乗りたいバスの経路も分かる。[5] これを図2.6に示す。

図2.6 バス NAVITIME

2.4 問題点

まず2.1節の Beacon は、GPS とは違い広範囲の利用はできない。屋内や地下で Beacon は使用することが可能だが常に広範囲を移動しているもの、今回ならば研究対象がバスなので屋外で使用できる GPS を使用しなければならない。2.2節のバス到着案内システムと2.3節のバス NAVITIME は、バス会社のサーバーにバス自体がどこにいるのかはわかるが、システムが大掛かりすぎる。さらに、発着時刻や、バスの経路までは情報として得ることができるが、リアルなバスの移動を視覚的に見ることはできない。
京都市内ではバス停にこれらのようなシステムが多く導入されているが全国すべてのバス停にあるわけではないため、利用に制限がある。

第3章　提案手法

2.4で示した問題点を解決するために、携帯端末用いて、わざわざ大掛かりなシステムを作らなくてもスマートフォンなどの携帯端末で解決でき、コスト的な削減もできるシステムを提案する。

3.1　提案手法

本研究では、路線バスの現在位置を推定するために、路線バスに乗車している不特定多数の乗客のアンドロイド端末からそれぞれの端末を識別できる情報とアンドロイド端末の位置情報と移動速度をサーバーに送信する。サーバー内で取得した位置情報と移動速度の近い値の端末をひとつのグループにまとめる。これらが予め決めておいた車の基準値と近い値ならば、これを路線バスと判断し、一定の間隔でそれぞれのアンドロイド端末にサーバーから路線バスの位置情報を返し、アンドロイド端末の画面上の GoogleMap にピンを表示するアプリケーションを提案する。
アンドロイド端末上のアプリケーションのアルゴリズムを図3.1(a)　図3.1(b)に示す。
サーバーアルゴリズムを図3.2に示す。

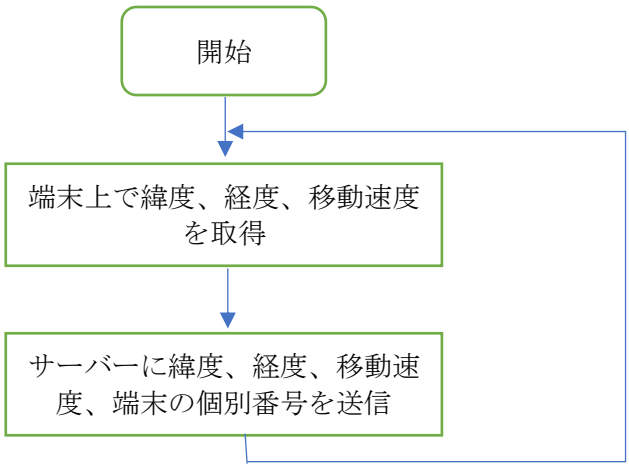


図3.1(a)　情報取得とサーバーへの送信

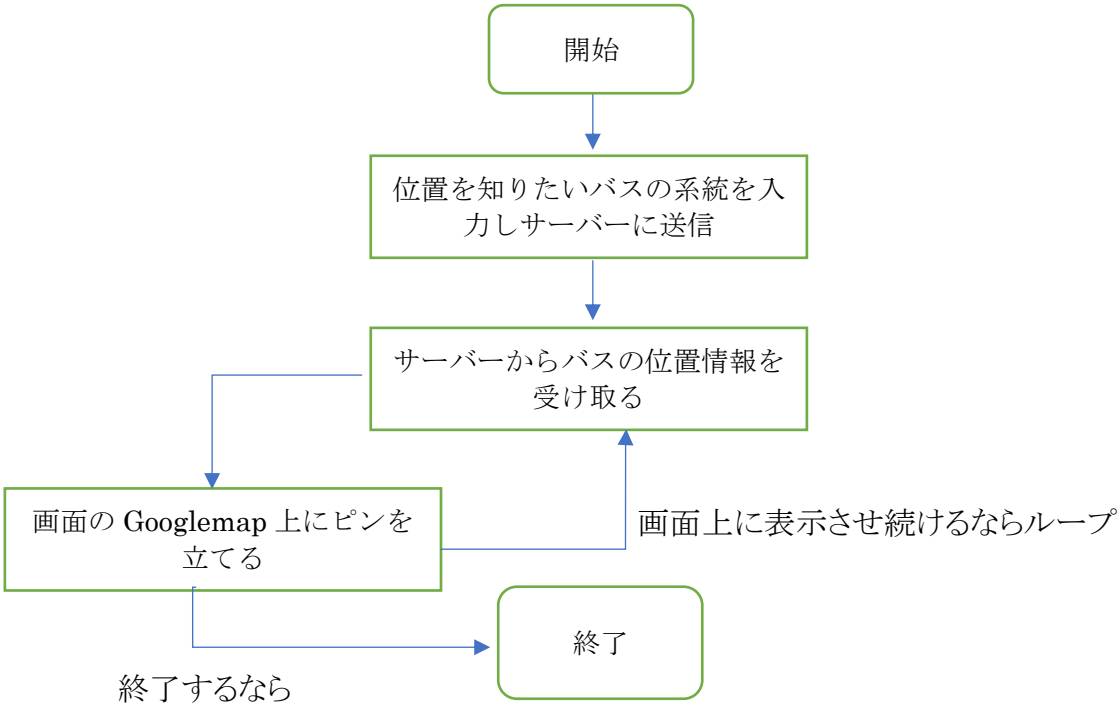


図3.1(b)　路線バスの位置表示

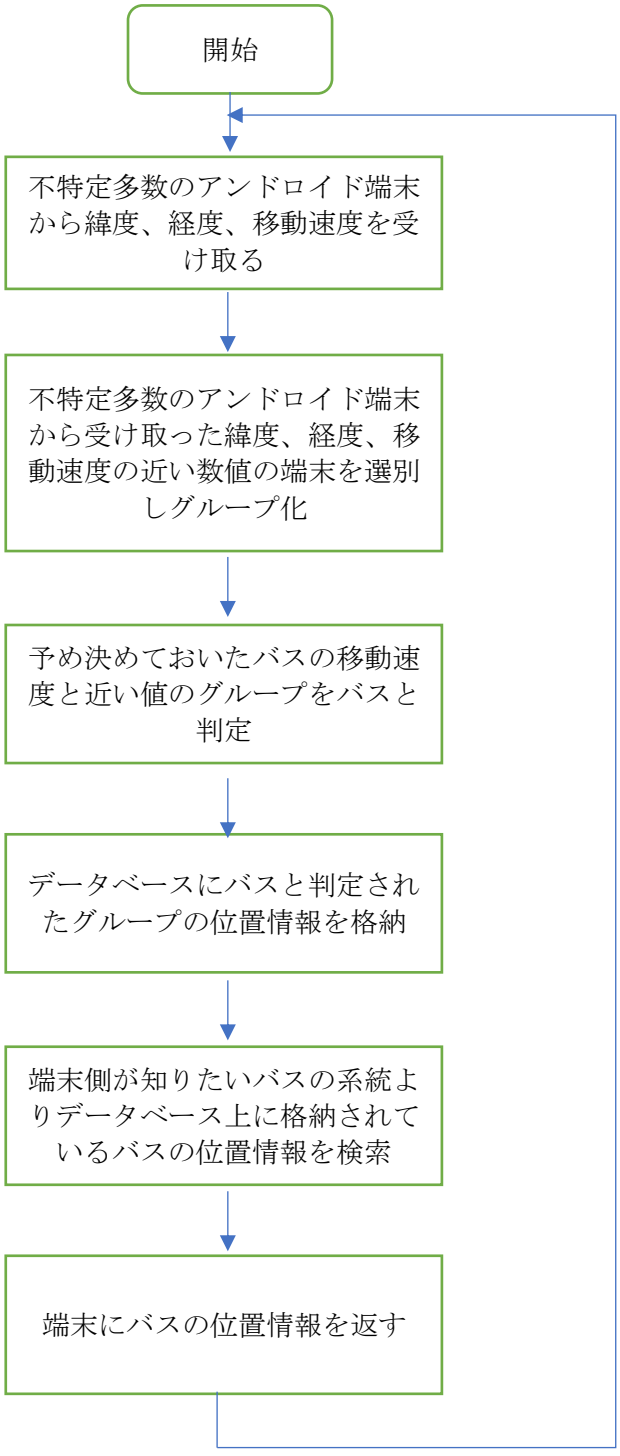


図3.2　サーバーアルゴリズム

第4章　実験

4.1　実験目的

本研究では、路線バスの動きを位置情報と移動速度から判断することができれば、提案手法におけるアルゴリズムを実現させることが可能だと考える。
提案手法による路線バスの位置情報を返すアプリケーションが実現可能である事を確かめるためテストアプリケーションを作成して様々なパターンから路線バスを特定できるか、その結果を確認する事を目的とする。

実験では、第3章の提案手法におけるアプリケーション側のアルゴリズム、サーバー側のアルゴリズムを実現させるためそれぞれ実験を行う。

4.2　実験1

(a)提案手法の図3.1(a)のアルゴリズムを実現させるために、アンドロイド端末で緯度、経度、移動速度を取得できることを確認するために、本実験では、自分の現在地の、緯度、経度そして、移動速度を画面上に表示し、三秒ごとにデータをテキスト形式のファイルに書き込みを行えることを確認する。

(b)提案手法の図3.1(b)のアルゴリズムを実現させるために、アンドロイド端末画面上にgooglemap を表示し、(a)で取得した緯度、経度のピンを立てられることを確認する。

4.2.1　実験1の結果

(a)アンドロイド端末で、緯度、経度、移動速度を取得し画面上に表示することができ、テキストファイルに結果を書き出すことができた。その図を図4.1に示す

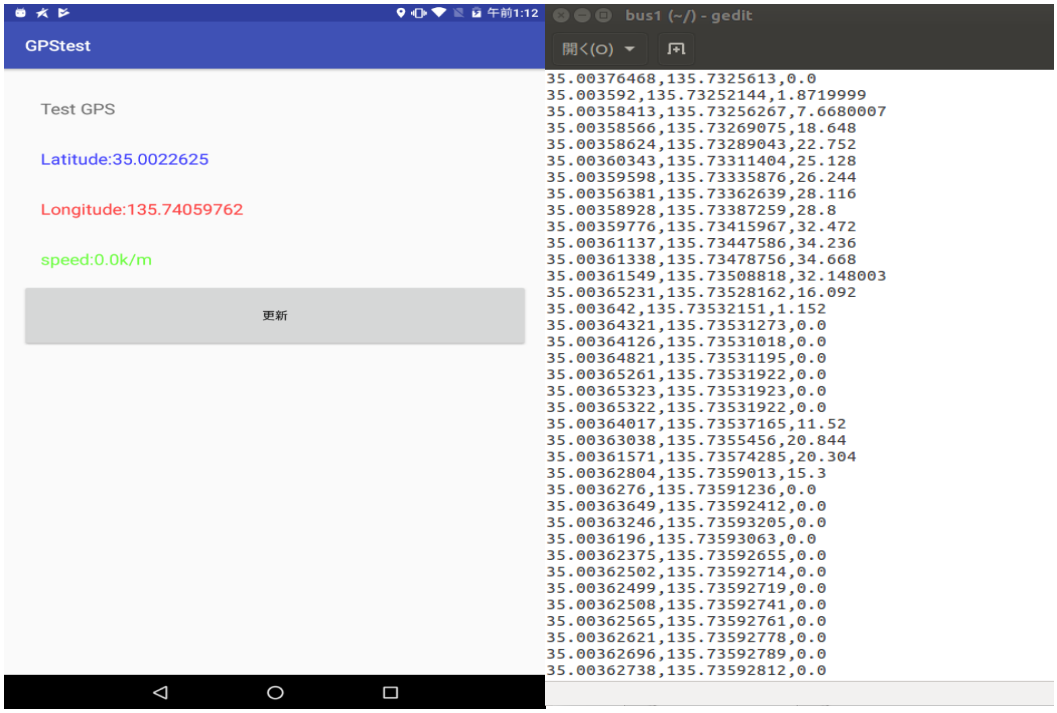


図4.1　端末上のアプリケーションの画面と出力したテキストファイル

今回の実験は、インターネット上の緯度と経度のみを取得できるサンプルアプリ[6]に速度を計測できる関数 getspeed()とファイルにテキストを出力できる savefile()を追加した。

(b) google map をアンドロイドアプリで使用するには、google から提供されている API[7]に用意されている関数 potision()に、緯度と経度を入力すると任意の場所にピンを立てることができることが分かった。その図を図4.2に示す。

図4.2　googlemap 上の任意の場所にピンを立てた図

自作アプリで、(a) (b)ともに動作することが確認できた。

4.3　実験2

アンドロイド端末から取得した情報をもとに、路線バスの位置を推定できるかを確認する実験を行った。

実験1で作成したアプリケーションを用いて実際に路線バスに乗り計測を行う。路線バスの移動を位置情報の値の変化で特徴をつかむことができれば路線バスかどうかの判定の基準値を決めることができる。この実験では、路線バスに乗車し実験1で作成したアプリケーションを起動する。このアプリケーションは三秒ごとに緯度、経度、移動速度を測定する。路線バスは十か所のバス停に止まるまで計測を行う。これを、同じルートで二回行う。

その後、一般車で路線バスと同じく、アプリケーションを起動し、路線バスが通ったルートで測定を行う。その際、40㎞を維持して走行を行う。

今回走行したルートは、西大路四条から東に向かって四条河原町まで走行し、その区間内のデータを100個記録し、緯度、経度、移動速度の3つのグラフを走行毎に作成し、その結果を考察する。

4.3.1 実験2の結果

路線バスの位置情報と移動速度の推移

実験から得られたデータのグラフを図4.3～4.8に示す。

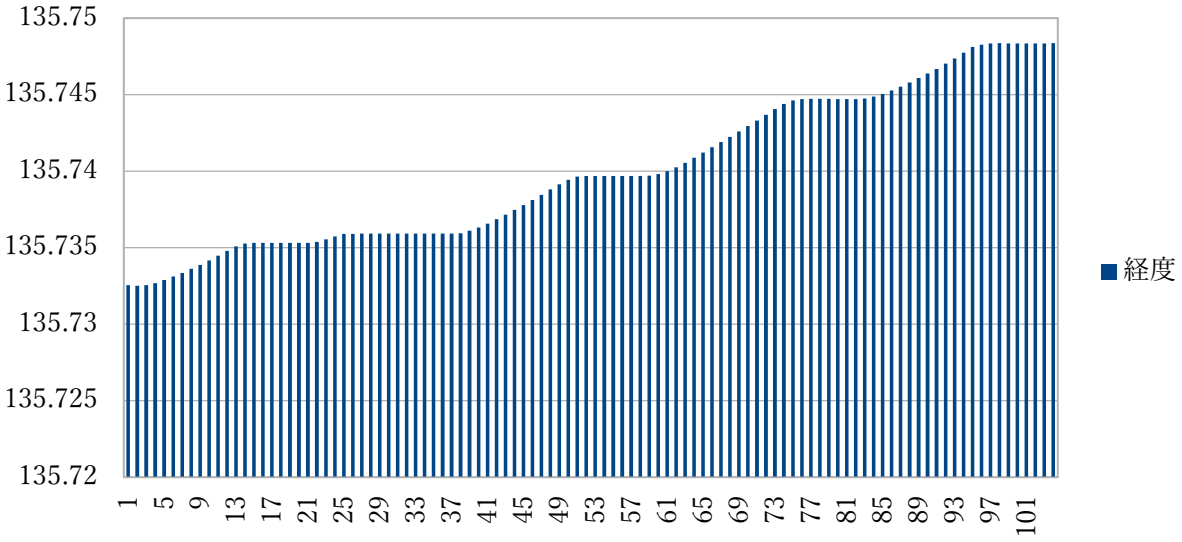


図4.3　路線バスの位置情報(経度)のグラフ

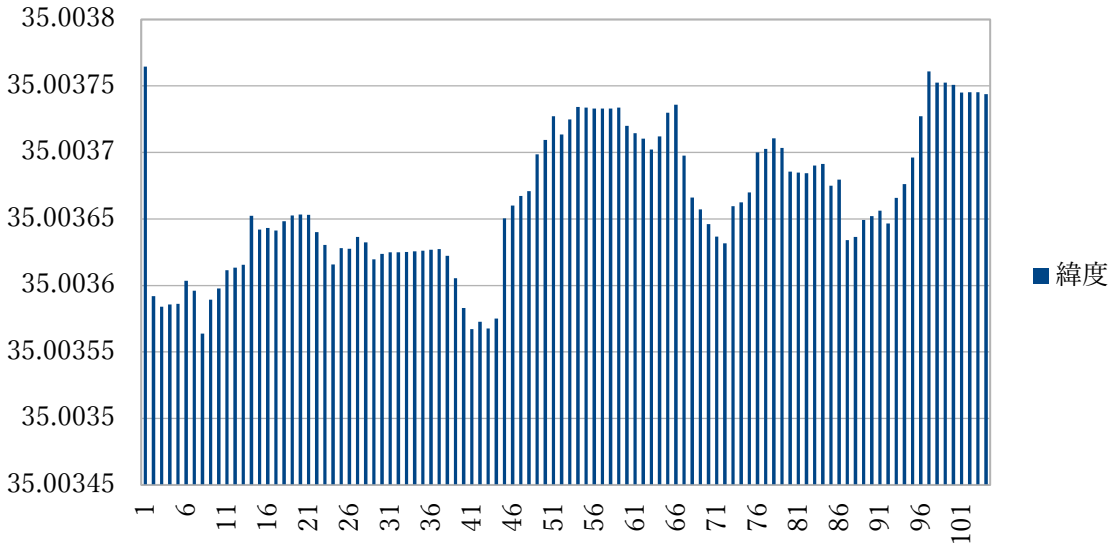


図4.4　路線バスの位置情報(緯度)のグラフ

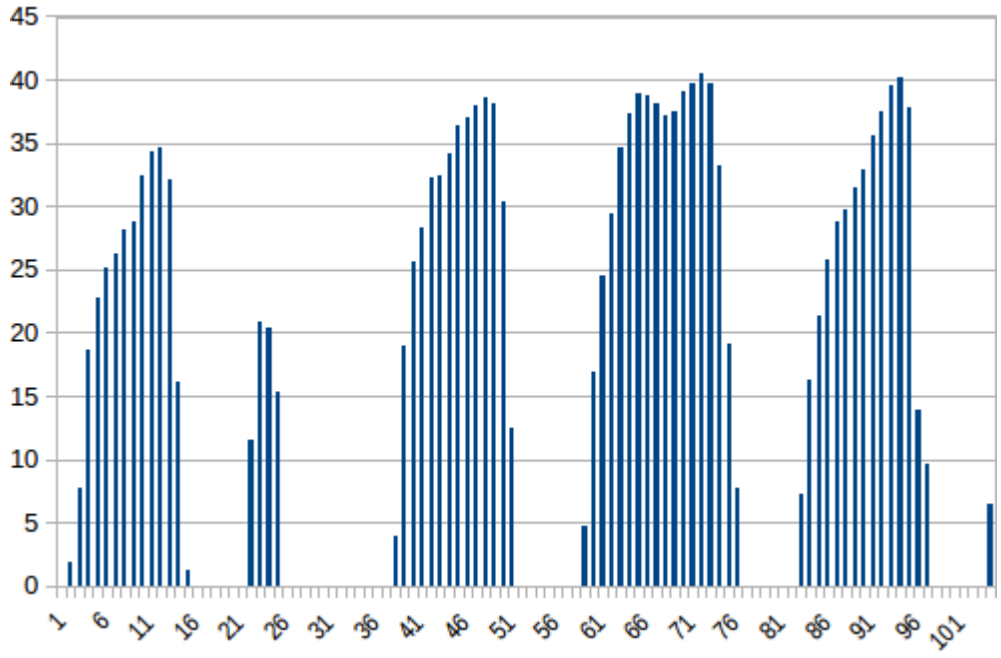


図4.5　路線バスの移動速度(k/m)のグラフ

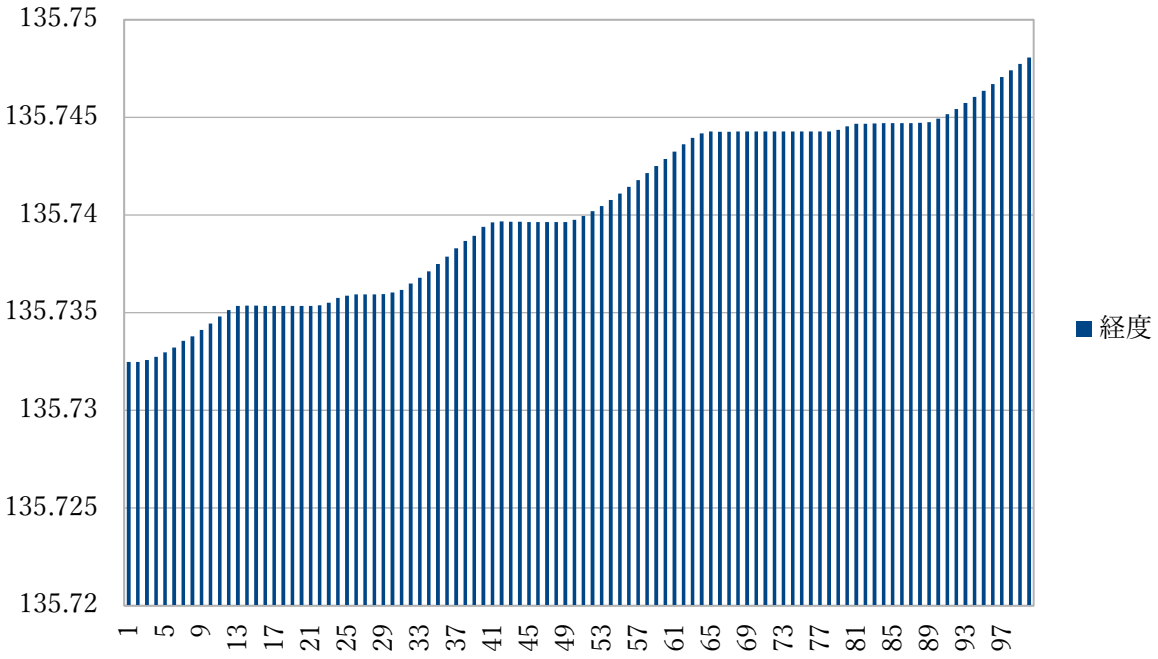


図4.6　路線バスの位置情報(経度)二回目のグラフ

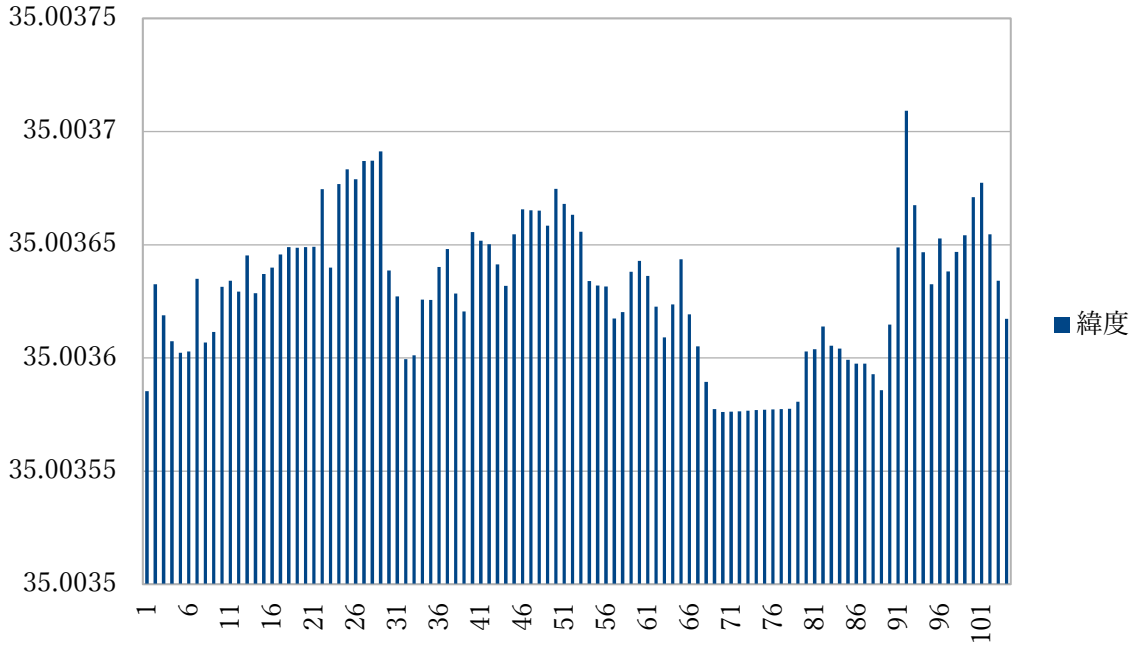


図4.7　路線バスの位置情報（緯度）二回目のグラフ

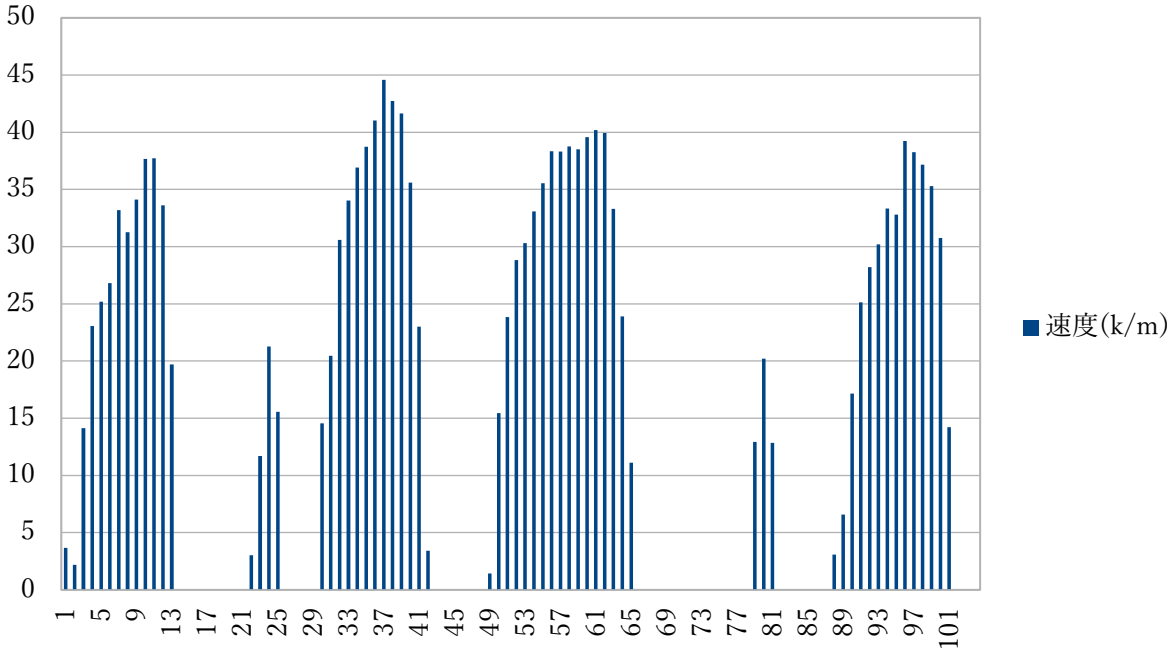


図4.8　路線バスの移動速度(k/m)二回目のグラフ

一般車の位置情報と移動速度の推移

実験から得られたデータを図4.9～4.11に示す。

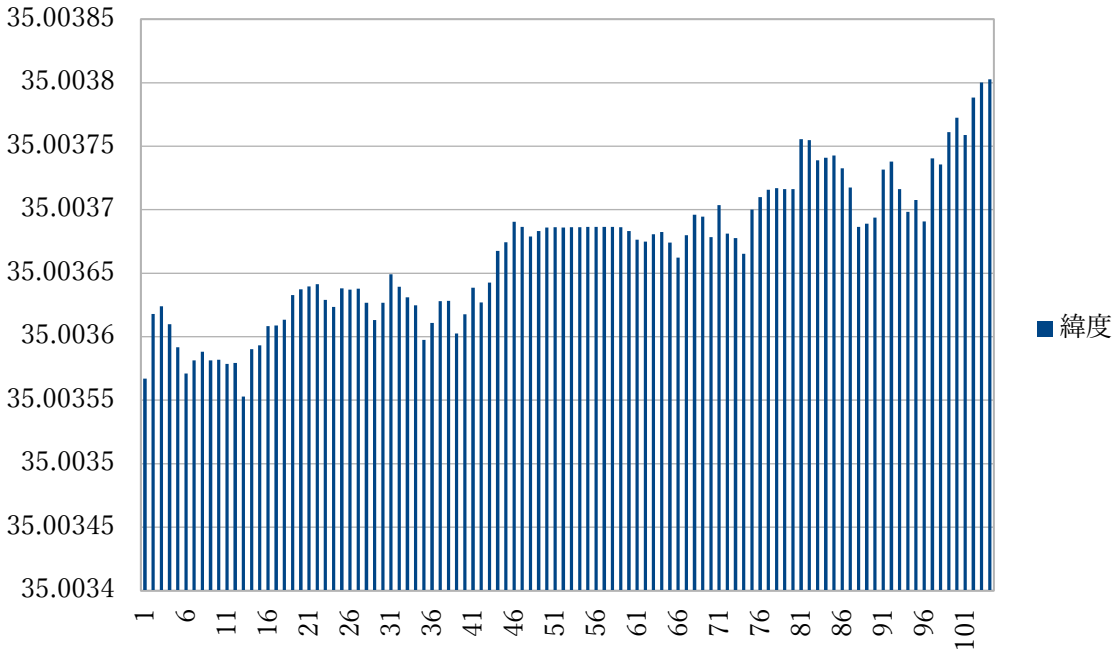


図4.9　一般車の位置情報（緯度）のグラフ

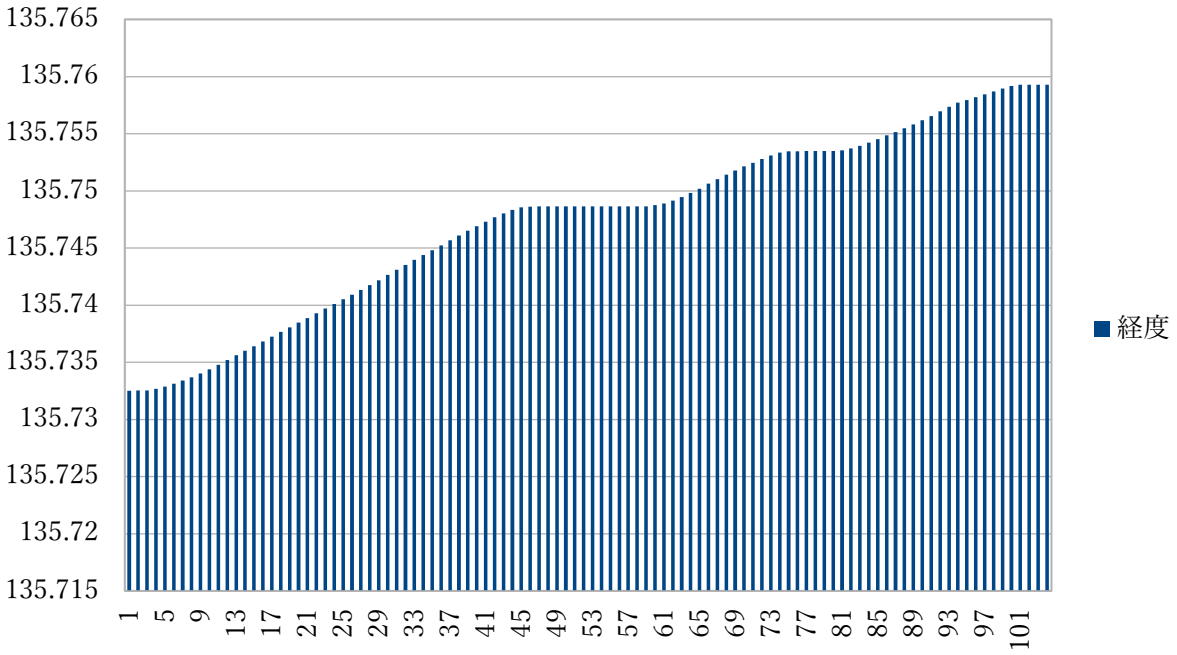


図4.10　一般車の位置情報（経度）のグラフ

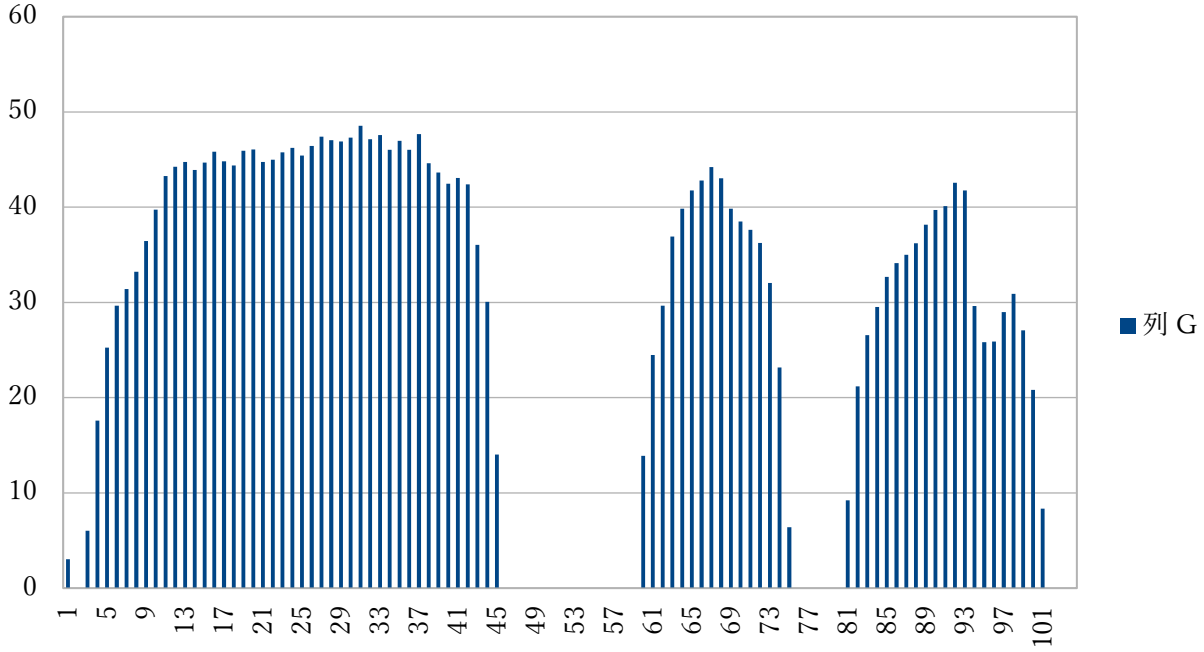


図4.11　一般車の移動速度のグラフ(k/m)

この実験は、西から東に進んだものである。そのため、一般車と路線バスどちらも図4.3、図4.6、図4.10から経度はデータ数が多くなるにつれて滑らかに高くなることが分かった。
路線バスの緯度は図4.4、図4.7からデータにばらつきが見られた。これは一般車の緯度のデータ図4.8が路線バスの緯度のデータより滑らかに数値が上昇していることから、路線バスはバス停に停車する際に歩道側によるため、路線バスの緯度は一般車に比べてデータにばらつきがあると考えられる。
また、経度・緯度に変化がないデータが横ばいの場合、停車している状態（歩道側あるいは車道側）である。移動速度については、路線バスの移動速度図4.5、図4.8から定期的な速度の減少がみられるため、路線バスはバス停で停車しているとわかる。一般車の移動速度、図4.11は路線バスに比べて停車の回数が少なく路線バスよりも速度が0になっていないので加速している時間が長いと推測できる。
以上のことから路線バスと一般車を区別するルールを考える。
今回の実験では、西大路四条の交差点から四条河原町の交差点まで、東に向かって走行したのでその区間の中から100個のデータを抜き出し区間を限定し、その緯度、経度の値から路線バスかどうかを判定したい集団がバス停が設置されている経路を走行しているかどうかが判断できる。
次に路線バスか一般車かの判定を行う。路線バスと一般車の移動速度のグラフより路線バスの移動速度は一般車に比べて、連続的に速度が時速０k/m になっている領域が多い。
路線バスの連続的に時速０k/m になっている区間は図4.5が5カ所、図4.8が6カ所、一般車の連続的に時速０k/m になっている区間は図4.11より3カ所。
路線バスと一般車のそれぞれの移動速度のグラフより連続的に時速0k/m の部分をカウントし、これを S とする。さらに移動速度のグラフの速度が時速0k/m ではない部分の値の平均をとり、V とする。
ここでルール1として、V＞=時速20k/m ならば、車両と判断する。
次に、ルール2として、0＜S＜＝4ならば一般車と判断する。
S＞5ならば、路線バスと判断する。
ルール1及びルール2を満たすことができれば、今回実験で走行したルートでは、路線バスを特定できると考えることができる。更に、長距離走行している車両を路線バスかどうか判定するには、取得した緯度、経度の位置情報から、バス停が設置されている経路上を走行しているかどうかをまず判定し、取得した全体の位置情報から100個のデータのまとまりをいくつか取り出し、上記のルールを用いて路線バスかどうかの判定を行うことができると考える。

第5章　まとめ

本研究では、アンドロイド端末の GPS センサーを用いて、路線バスの位置情報を視覚化するシステムを提案した。
路線バスの現在位置を表示するアプリを自作し実験を行い、うまく動作することを確認した。
今回サーバとの通信部分と開発と動作確認を行う動作確認をするところまで実験することはできなかった。
路線バスと一般車区別するためのアルゴリズムを評価するために、実際に路線バスに乗車し、路線バスの位置情報と移動速度から、路線バスの動きを確認する実験を行った。
渋滞を考えないとするならば、本研究では、一般車と路線バスの区別することができると考えられる。しかし実験では、データの変動が時間帯や交通状況が影響するので、その点を考慮しきれなかったという反省点がある。完全に路線バスを特定するには、本研究よりさらに複雑な制限下のもとで実験を行いデータを取る必要性がある。

謝辞

本研究を進めるにあたりご指導を頂いた卒業論文、指導教員の三好 力教授に感謝いたします。また、日常の議論を通じて多くの知識や示唆を頂いた三好研究室の皆様に感謝します。

参考文献

[1]. 総務省
<http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h28/html/nc252110.html>
[2]. 東洋経済オンライン
<http://toyokeizai.net/articles/-/113807>
[3]. Beacon とは？
<http://iot-jp.com/iotsummary/iottech/bluetooth/beacon%E3%81%A8%E3%81%AF%EF%BC%9F/.html#iBeacon-2>
[4]. IT メディアエンタープライス
<http://www.itmedia.co.jp/enterprise/articles/1501/06/news093.html>
[5]. バス NAVITIME
<https://www.navitime.co.jp/bustransit/>
[6]. [Android] GPS で位置情報を取得するアプリを作る
<https://akira-watson.com/android/gps.html>
[7]. Google maps API
<https://developers.google.com/maps/?hl=ja >

付録１　位置情報取得アプリソースコード

```
package com.example.kz_pc.gpstest;
import android.Manifest;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Criteria;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.location.LocationProvider;
import android.os.Environment;
import android.provider.Settings;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
import android.util.Log;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.text.SimpleDateFormat;
import java.util.Date;
public class MainActivity extends AppCompatActivity implements
LocationListener {

    private LocationManager locationManager;
    private String filename = "testgps.txt";
    private int count = 1 ;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button btn = (Button)findViewById(R.id.button);
        // クリックイベントを受け取れるようにする
        btn.setOnClickListener(new View.OnClickListener() {
            // このメソッドがクリック毎に呼び出される
            public void onClick(View v) {
                // ここにクリックされたときの処理を記述
                    locationStart();
            }
        });
    }
    private void locationStart(){
        Log.d("debug","locationStart()");
        // LocationManager インスタンス生成
        locationManager = (LocationManager)
getSystemService(LOCATION_SERVICE);
        final Boolean gpsEnabled =
locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER
);
        if (!gpsEnabled) {
            // GPS を設定するように促す
            Intent settingsIntent = new
Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
            startActivity(settingsIntent);
            Log.d("debug", "not gpsEnable, startActivity");
        } else {
            Log.d("debug", "gpsEnabled");
        }
        if(ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)!=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION,}, 1000);
            Log.d("debug", "checkSelfPermission false");
            return;
        }
locationManager.requestLocationUpdates(LocationManager.GPS_PRO
VIDER, 1000, 50, this);
    }
    // 結果の受け取り
    @Override
    public void onRequestPermissionsResult(int requestCode,
String[] permissions, int[] grantResults) {
        if (requestCode == 1000) {
            // 使用が許可された
            if (grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                Log.d("debug","checkSelfPermission true");
                locationStart();
                return;
            } else {
                // それでも拒否された時の対応
                Toast toast = Toast.makeText(this, "これ以上なにもできません", Toast.LENGTH_SHORT);
                toast.show();
            }
        }
    }
    @Override
    public void onStatusChanged(String provider, int status, Bundle
extras) {
        switch (status) {
            case LocationProvider.AVAILABLE:
                Log.d("debug", "LocationProvider.AVAILABLE");
                break;
            case LocationProvider.OUT_OF_SERVICE:
                Log.d("debug",
"LocationProvider.OUT_OF_SERVICE");
                break;
            case LocationProvider.TEMPORARILY_UNAVAILABLE:
                Log.d("debug",
"LocationProvider.TEMPORARILY_UNAVAILABLE");
                break;
        }
    }
    @Override
    public void onLocationChanged(Location location) {
        String text = String.valueOf(location.getLatitude()) + ",";
        String text2 =String.valueOf(location.getLongitude()) + ",";
        float sokudo = location.getSpeed();
        String speed = String.valueOf(sokudo*3600/1000) + "¥n";
        // 緯度の表示
            TextView textView1 = (TextView)
findViewById(R.id.text_view1);
            savefile(filename, text);
            textView1.setText("Latitude:"+location.getLatitude());
            Log.v("緯度 ", String.valueOf(location.getLatitude()));
            // 経度の表示
            TextView textView2 = (TextView)
findViewById(R.id.text_view2);
            savefile(filename, text2);
          textView2.setText("Longitude:"+location.getLongitude());
            Log.v("経度", String.valueOf(location.getLongitude()));
            TextView textView3 = (TextView)
findViewById(R.id.text_view3);
textView3.setText("speed:"+location.getSpeed()*3600/1000 +
"k/m");
       savefile(filename, speed);
        count++;
        }
    public void savefile(String file, String str){
        FileOutputStream fileOutputStream = null;
        try {
            fileOutputStream = openFileOutput(file,
Context.MODE_APPEND);
            fileOutputStream.write(str.getBytes());
        }catch (IOException e) {
            e.printStackTrace();
        }
    }
    @Override
    public void onProviderEnabled(String provider) {
    }
    @Override
    public void onProviderDisabled(String provider) {
    }
}
```

付録２　googlemap 表示アプリソースコード

```
package com.moonlight_aska.android.googlemapv2;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.CameraPosition;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import android.os.Bundle;
import android.support.v4.app.FragmentActivity;
public class MainActivity extends FragmentActivity {
    // 六甲山：北緯 34 度 46 分 41 秒, 東経 135 度 15 分 49 秒}
    private double mLatitude = 34.0d + 46.0d/60 + 41.0d/(60*60);
        private double mLongitude = 135.0d + 15.0d/60 +
49.0d/(60*60);
    private GoogleMap mMap = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
mMap=
(  (SupportMapFragment)getSupportFragmentManager().findFragme
ntById(R.id.map) ).getMap();
        if (mMap != null) {
            LatLng location = new LatLng(mLatitude, mLongitude);
            CameraPosition cameraPos = new CameraPosition.Builder()
                .target(location).zoom(10.0f)
                .bearing(0).build();
mMap.animateCamera(CameraUpdateFactory.newCameraPosition(ca
meraPos));
            // マーカー設定
            MarkerOptions options = new MarkerOptions();
            options.position(location);
            mMap.addMarker(options);
        }
    }
}
```