

GPS 付き携帯電話を用いた位置情報の活用

Practical use of Position Information using the Cellular Phone with GPS

上石 剛浩
Takehiro KAMIISHI

明治大学大学院 理工学研究科 基礎理工学専攻 ソフトウェア基礎研究室
(〒214-8571 神奈川県川崎市多摩区東三田 1-1-1 E-mail: kamiishi@cs.meiji.ac.jp)

ABSTRACT. Since a cellular phone can treat position information now, very convenient service can be offered. However, in the present condition, in order to offer position information service, very much knowledge and cost are required. Then, we create the "position information framework" which can make position information service easily. moreover, we create the "GPS emulator" and "social information emulator" simulating the technology of the very near future. Based on these, by evaluating and building some position information services, we search for whether offer of what position information service is attained and look for what position information service is useful.

1. 背景

近年の情報社会の中で、携帯電話の発展は目覚しく、また、日本が世界に先駆けている数少ない分野の一つでもある。中でも、携帯電話が位置情報を扱えるようになったことは、多くの人々が、自分の位置情報をサービスプロバイダに提供できるようになったことを意味し、社会システムに位置情報を組み込むことで、より便利な社会になる可能性も秘めている。

しかしながら、位置情報を扱ったサービスを提供するためには、その土台として、多くの位置情報に関する知識や、複数の国際規格に精通する必要があり、低いコストで高品質なサービスを素早く提供することの妨げになっている。

2. 目的

そこで、本論文では、位置情報を用いたサービスを容易に提供できるようにするために、位置情報のフレームワークを構築すると共に、いくつかの仮定の下で、実際に位置情報サービスの実験を行うことによって、ごく近い未来の技術の上で可能になるとされる位置情報のサービスの可能性を探るものである。

具体的には、様々な GPS 機器から接続されることを前提に設計されている既存の多くの位置情報プラットフォームと異なり、携帯電話に付属の GPS 機器にのみ着目して設計することで、携帯電話付属の GPS から得られる位置情報を、特別な知識なしにサービスに結び付けられるようになっている『位置情報フレームワーク』を作成する。これにより、位置情報を利用したサービスが、位置情報に関する専門的な知識なしに、簡単に提供できるようになる。また、ごく近い未来に携帯電話に搭載されるであろう GPS をシミュレートする『GPS エミュレーター』と、ごく近い未来に、ITS(高度交通システム)等によって、誰もが入手可能になるであろう社会的な情報をシミュレートする『社会情報エミュレーター』を作成することで、

ごく近い未来の技術の上で、どのような位置情報サービスが提供可能になり、どのような位置情報サービスが有用であるのか探求する。

3. 本システムの構成

(1) 全体の構成

本論文の目標を達成するために、以下の構成からなるシステムを構築した。

- 携帯電話の位置情報にのみ着目した、位置情報フレームワーク
- 現在、携帯電話に用いられているより、精度の高い GPS をシミュレートする GPS エミュレーター
- 日常一般に得られる情報をシミュレートする社会情報エミュレーター

これらの構成図は、以下の図 3.1 の通りである。

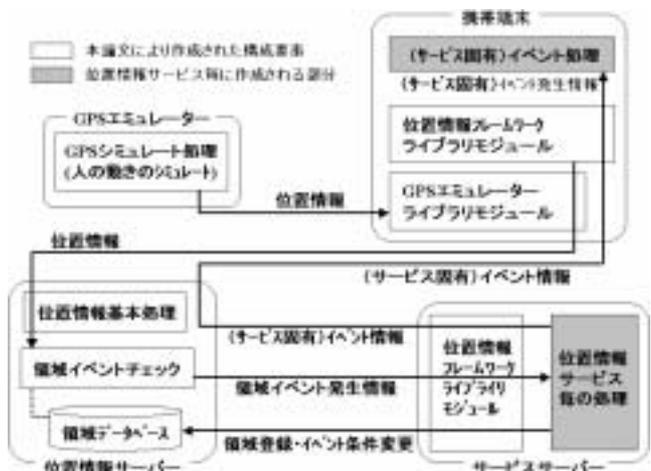


図 3.1 本システム全体の構成

なお、構成要素の一つである社会情報エミュレーター

は、位置情報サービスの補助として、様々な社会的情報(交通情報・地図情報等)をシミュレートするものであるため、構成図上は、サービスサーバー内に区分され、この構成図には表れてこない。

また、この図を見ると分かるように、本論文のフレームワークは、後述する領域という概念を基本にしており、この領域の概念によって位置情報を覆い隠すことで、位置情報に関する知識や変換処理などを不要にしている。

(2) 位置情報フレームワーク

本論文で提案するフレームワークは、位置情報を自動的に『領域』という概念に変換することが最大の特徴である。この領域に関する特徴は次節で詳しく述べられているので、そちらに譲るとして、ここでは、それ以外の特徴を挙げてみる。領域以外の大きな特徴としては、以下のような点が挙げられる。

- 位置情報基礎変換機能
- 簡易 HTTP サーバー機能
- ユーザー情報の保持と計算機能

まず、位置情報基本変換機能であるが、これは、GPSの規格ごとの差異を吸収したり、測位方法による位置情報のずれなどを自動的に変換し、1つの位置情報サービスに対して、異なる規格のGPS機器を搭載する携帯端末が接続をしたとしても、それらの違いを考慮することなく、サービスを提供できるようにする機能である。

次に、簡易 HTTP サーバー機能だが、これは、多くの携帯端末の制限から必要になった機能である。一般的に携帯端末では、TCP/IPによる自由な通信が許可されおらず、携帯端末側から任意のHTTPサーバーへのHTTPリクエストのみが許可されている。このため、任意のサーバー側から任意のタイミングでサービス固有のデータを携帯端末に送ることは、基本的には不可能である。しかし、位置情報サービスを構築するためには、任意のタイミングで任意のデータをサーバー側から携帯端末に送る必要が出てくるため、提案するフレームワークでは、簡易 HTTP サーバー機能を搭載し、携帯端末からこの簡易 HTTP サーバーへアクセスさせるようにし、さらにそのHTTP通信の上に、独自のプロトコルを制定することで、この問題を解決している。

最後のユーザー情報の保持と計算機能とは、ユーザーから送られてくる位置情報を保存し、その位置情報から算出される情報を計算する、ということである。位置情報を扱わずに位置情報サービスを構築できるようにするためには、領域という概念で位置情報を覆い隠すだけでなく、ユーザーの現在の速度など、過去から現在への連続した位置情報から計算可能な情報も位置情報サーバー上で自動的に計算しなければならない、ということである。よって、提案するフレームワーク上では、ユーザーの位置情報を保存し、そこから算出される情報を自動的に計算するようになっている。

(3) 領域

領域とは、平面を表す頂点集合と高さ情報によって表される任意の柱体である。サービスにとって意味のある現実世界の任意の空間を領域という概念で表わし、数字の羅列である緯度・経度ではなく、この領域情報によってサービスを構築することにより、位置情報を直接扱う場合に要求される様々な知識や多くの変換処理などからサービス提供者は解放されるのである。

本論文の領域とは、以下に説明される通りのものである。

- 領域 ID によって区別される
 - 領域 ID を用いて親を指定することにより、領域のツリーを構築できる
 - 平面を表す情報と高さ情報によって決まる柱体をその領域の範囲とする
 - ユーザーが領域の境界をまたいだ場合、又は、内部で移動した場合に領域イベントを発生できる
 - 領域イベントは、ユーザーの制限や発生するイベントの種類を指定できる
 - 親の領域のイベント発生条件が満たされない場合、子の領域のイベント発生チェックは行われない
 - 子の領域でイベントが発生した場合、親の領域のイベント発生は無視される
- 具体的な例を挙げて説明すると、例えば、4章1節で説明されるオリエンテーリングサービスは
- 「登録所」でユーザー登録をする
 - 「スタート地点」に入った瞬間から「ゴール地点」に入る瞬間までの時間を競う
 - 途中、いくつかの「チェックポイント」に立ち寄りなければならない
 - 途中、「ボーナスポイント」に立ち寄った場合には、ボーナスが与えられる

というサービスであるが、このサービスで本質的に必要な情報は、GPSから直接得られる緯度経度といった位置情報ではなく、ユーザーが現在「登録所」や「スタート地点」といった場所のうち、どの場所にいるか、という情報である。つまり、これら「登録所」や「スタート地点」を領域という情報で表すことができれば、後は、位置情報サーバー上で、ユーザーから送られてきた位置情報を領域情報へと自動的に変換してくれるため、サービスサーバー上では、位置情報を直接扱う必要は全くなく、領域情報のみを扱えばいいようになるのである。

このようにサービスにとって重要な場所の情報を領域によって表すことで、サービス提供者は、位置情報を扱う必要がなく、位置情報に関する特別な知識が要求されたり、位置情報に関する処理をすることなく、誰でも気軽に位置情報を用いたサービスを提供できるようになる。

(4) GPS エミュレーター

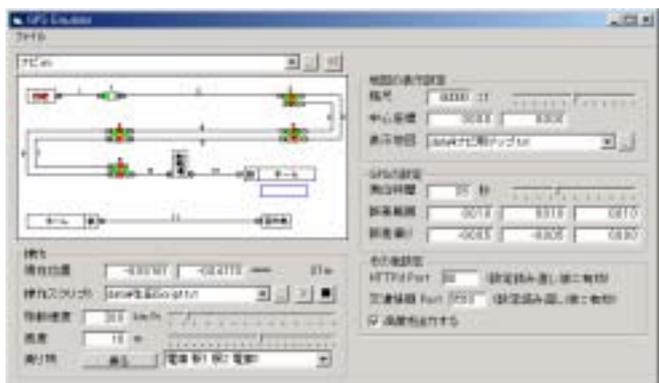


図 3.2 GPS エミュレーター

GPS エミュレーターは、将来携帯電話に搭載されるであろう精度の高いGPSをエミュレートするものであり、その本質は、人の動きをシミュレートすることにある。現在でも、GPSの代わりにするGPSエミュレーターは存在するが、その多くは、ただ単に数値を入力し、その数値がGPSから出力される情報の代わりとなるだけのものばかりである。しかし、本論文で作成したGPSエミュレ

ーターは、人の動きをシミュレートし、その人の動きの結果としての連続した位置情報を GPS から出力される情報の代わりとするものである。具体的には、本論文で作成した GPS エミュレーターは以下のような機能・特徴を持っている。

- 画面上をクリックするだけで、その方向へ人が簡単に動くようにできている
- 人は、現在位置・高度・移動速度によって表わされ、これらを簡単に操作することができる
- 人の動きは、スクリプトとして表現することができ、自動で制御することもできる
- 画面上に任意の地図を表示することができる
- GPS の精度として、測位間隔・誤差・偏りを設定することができる
- 社会情報エミュレーターと連動し、信号機や乗り物の情報を表示し、乗り物に乗ることができる

このような機能・特徴を持つ GPS エミュレーターにより、精度の高い GPS が携帯電話に搭載されたと仮定したときに、果たしてどのような位置情報サービスが提供可能で、どのような位置情報サービスが有用であるのかの検証をすることが可能となる

(5) 社会情報エミュレーター

ITS(高度交通システム)に代表されるような交通情報や付近の地図情報などは、近い将来に情報化社会の一部として組み込まれることが十分に予想され、将来は、このような情報によって、社会サービスは支えられるものと思われるが、これらの情報を活用したサービスの研究は、関連理論こそ研究されているものの、実際のサービスの研究はあまりされていない。これには、実際のサービスの研究には、大規模な施設などが必要になってしまうこと等が理由に挙げられるが、この問題を解決し、より多くの人間が社会情報を利用した位置情報サービスの研究を可能にするためにも、本論文では、社会的な情報を計算機上でシミュレートする社会情報エミュレーターを作成した。このエミュレーターは、主に、位置情報サービスと深い関係を持つと思われる交通情報をシミュレートするものである。具体的には、交通情報を

- 信号機のように、決まった時間毎に ON/OFF を繰り返すもの
- バスや電車のように、決まった時刻にある地点を出発し、目的地に向かうものに分け、これらの交通情報を設定に従ってシミュレートし、いつでもその情報を取り出せるようにすることで、将来における、誰でもが社会的な情報を利用したサービスを気軽に構築することができる、という状況を作り出している。これにより、本論文の目標である、近い将来における位置情報サービスの探求が可能となる。

4. 位置情報サービスの例と評価

本論文では、3章で述べた位置情報フレームワークと各エミュレーターを用いて、近い将来に可能になるであろうと思われる位置情報サービスを探求するのが目的であるが、その前に、いくつかの位置情報サービスを、提案する位置情報フレームワーク上で実装し、このフレームワークの評価を行った。以下で、それらの詳細を述べつつ、評価を記した後で、近い将来に可能になるであろう位置情報サービスの探求の結果について述べる。

(1) オリエンテーリングサービス

オリエンテーリングサービスは、提案するフレームワ

ークの基本的な全ての機能をチェックするためのサービスとして作成したサービスであり、具体的には、次のようなサービスある。

- 「登録所」でユーザー登録をする
 - 「スタート地点」に入った瞬間から「ゴール地点」に入る瞬間までの時間を競う
 - 途中、いくつかの「チェックポイント」に立ち寄らなければならない
 - 途中、「ボーナスポイント」に立ち寄った場合には、ボーナスが与えられる
- これら説明のうち「」で囲まれた中は、領域として扱われ、矩形情報を持たない整理用の領域と併せて、以下の図 4.1 のような領域ツリーとして位置情報サーバーに登録される。



図 4.1 オリエンテーリングサービスの領域ツリー

このオリエンテーリングサービスにおいて、チェックポイントが二つの場合の動作画面を図 4.2 に示す。



図 4.2 オリエンテーリングサービスの実例

この例を通して、提案する位置情報フレームワークの以下の機能が正常に動作していることを確認した。

- 位置情報サービスの登録と削除に関する全ての機能
- 領域に関する全ての機能
- ユーザーIDの自動生成とユーザーの登録・削除に関する全ての機能
- ユーザーの位置情報から登録された領域情報への

変換に関する全ての機能

● サービス固有データの送受信に関する全ての機能

以上の点から、提案する位置情報フレームワークは、ほぼ全ての機能が正常に動作し、位置情報サービスのフレームワークとして十分に機能していることが確認できた。

この例を通して得られた評価としては、オリエンテーリングサービスのような領域ツリーを構築し、位置情報サービス上で、位置情報を自動的に領域情報へと変換するということは、ユーザーから送られてくる位置情報のうち、特に意味のない位置情報を自動的に切り捨て、領域の境界線をまたいだ場合の位置情報のみを取得できるということであり、これは、その位置情報サービスにとって、本質的に必要な位置情報のみに着目することができ、本質的に必要な処理のみを行うことができるということである。そして、このオリエンテーリングサービスでは、位置情報を扱うような処理は全く行われておらず、領域に関する処理のみでサービスが実現されている。つまり、位置情報に関する特別な知識なしに、誰でも気軽に位置情報サービスが構築できる、という目標をこのサービスに限って言えば、実現できているのである。

ただし、当然のことではあるが、これは、オリエンテーリングサービスで本質的に重要である位置情報が領域という概念で表される情報に綺麗に変換できるために、位置情報を全く扱うことなく、位置情報サービスが提供できているのだと予想される。このため、本論文では、後述する位置情報追跡サービスというサービスを通して、位置情報を直接扱わなければならない位置情報サービスについても評価を行っている。

また、位置情報を全く扱わなくても位置情報サービスが提供できているとはいっても、位置情報の集合によって表される領域情報は予め作成しておかなければならず、この領域情報作成の際には、位置情報を扱わなければならない。ただし、この問題点も、携帯端末のボタンを押した場所を頂点とする領域を自動的に作成し、出力するような領域作成ツールを作成することで、位置情報を扱うことから解放されると考えられる。

(2) 仮想掲示板サービス

次に、提案する位置情報フレームワークを使用しない場合と使用する場合で、サービスサーバーのコーディングに要する労力の差がどの程度出てくるのか、仮想掲示板サービスというサービスを通して検証した。具体的には、次のようなサービスである。

- 現実世界にマッピングされる掲示板サービス
- 掲示板は、書き込まれている内容と掲示板が存在している位置を情報として持つ
- 掲示板を読み書きするためには、その掲示板が存在している場所へ行かなければならない

つまり、例えば、ある特定の部屋に関する掲示板を作成したければ、その部屋を表す位置情報の集合を掲示板の書き込み可能な範囲の情報として持ち、その部屋の中にいるときにのみ読み書きできる掲示板を仮想空間上に作成できたり、ある特定のパソコンの半径1m以内のみ読み書きできる掲示板を仮想空間上に作成できたりするようなサービスである。

このサービスは、提案するフレームワーク上で、掲示板の読み書き可能な範囲が領域として表され、その領域の中にいる場合のみ、対応する掲示板に読み書きできるようになっている。

この仮想掲示板サービスを実際に使っている例を図

4.3 に示す。



図 4.3 仮想掲示板サービスの実例

このサービスでは、掲示板の読み書き可能な範囲が、領域として綺麗に表現できているため、位置情報に関する処理をすることなしに、仮想掲示板サービスが実現できている。では、実際に、この仮想掲示板サービスを実現するためには、提案するフレームワークを利用する場合としない場合で、どの程度労力に差が出るのか評価してみたいと思う。この評価には、様々な方法が考えられるが、ここでは、客観的に評価できる方法として、それぞれのサービスサーバーでのソースコードの行数を比べてみたいと思う。

まず、提案するフレームワークを使用しない場合のサービスサーバーのソースコードの行数であるが、これは、全体でおおよそ500行程度であった。これに対して、提案するフレームワークを使用した場合のサービスサーバーのソースコードの行数の内訳は、仮想掲示板のサービスの本質的な処理に関する部分が約200行程度であり、位置情報フレームワークの通信やコマンドをラッピングするライブラリモジュールの部分が約450行程度であった。このうち、位置情報フレームワークのライブラリモジュールについては、仮想掲示板サービスのサービス提供者は作成する必要がないため、位置情報フレームワークを使用した場合に、仮想掲示板のサービス提供者が作成しなければならないソースコードの行数は、約200行程度のみ、ということになる。位置情報フレームワークを使用した場合、位置情報フレームワークのライブラリモジュールを用いることで、通信に関する処理や位置情報に関する処理が不要になるため、位置情報フレームワークを使用しない場合の500行程度に比べて、ちょうど位置情報フレームワークのライブラリモジュールの行数分の労力が不要になり、結果として、200行程度の労力で、仮想掲示板というサービスが構築できているのではないかと考えられる。

また、興味深い1つの事実としては、ネットワーク上での通信機能がプログラミング言語のライブラリの機能として与えられている言語の場合、簡易掲示板システムを作成すると、大体100行~400行程度であることが多い。これは、掲示板システムを構築するための環境整備に関する部分を除いた、掲示板システムの本質的なサービス処理をする部分のソースコードが、大体どのような言語でも100行~400行程度で表現されることを示しており、その意味でも、提案する位置情報フレームワークを用いた仮想掲示板が200行程度で表現できているということは、位置情報フレームワークによって、仮想掲示板に、本質的に必要な処理以外の環境作りができてい

という事実の1つの証明になっているのではないかとと思われる。

(3) 位置情報追跡サービス

次に、位置情報を直接扱わなければならないような位置情報サービスでは、提案するフレームワークを使用した場合に、どの程度有用であるのかを位置情報追跡サービスというサービスを通して検証した。具体的には、次のようなサービスである。

- ユーザーが特定の時間内に移動した経路情報を保存するサービス
- ユーザーの現在の移動経路もリアルタイムで表示・保存する
- ユーザーと簡単なコミュニケーション(会話)が取れるようにする

例えばこれは、運送会社のトラックが過去の特定の時間にどの場所にいたのかを表示・保存し、また、現在の経路をリアルタイムで表示・保存をするサービスで、必要であれば、そのトラックの運転者と会社のオペレーターの間で会話をする事ができるようなサービスである。

この位置情報追跡サービスの位置情報追跡機能一例を図4.4に、コミュニケーション機能の一例を図4.5に示す。

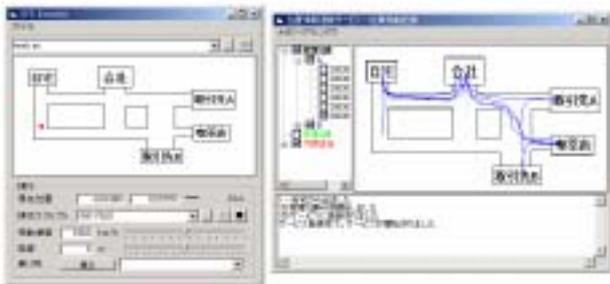


図 4.4 位置情報追跡機能の一例



図 4.5 コミュニケーション機能の一例

このサービスでは、これまでの位置情報サービス例とは異なり、サービスにとって意味のある位置情報を領域という概念で表すことができない。また、経路の軌跡を保存するという事は、このサービスにとって本質的に意味のある位置情報が、携帯端末から送られてくる位置情報全てであるということでもあり、提案するフレームワーク上での領域という概念によって、サービスにとって本質的な位置情報のみを選別し、サービスにとって本質的な処理のみを行う、ということが全くできないサービスとなっている。実際、このサービスのサービスサーバープログラムでは、位置情報に関する知識を用いて、位置情報を直接扱い処理するコードが存在している。これは、提案するフレームワークを使用することで、位置情報に関する特別な知識なしに、誰でも気軽に位置情報サービスを提供することができるようにする、という本論文の目標の1つに大きく反している。この問題点を解決するためには、いくつかの方法が考えられると思うが、このサービスでは「ユーザーの移動の軌跡を表示する」

という処理がどうしても必要になってしまうため、どのような方法をとったとしても、結局は、位置情報に関する何らかの処理が必要になってしまうのではないかと予想される。であるならば、その位置情報に関する何らかの処理をなるべく簡単に記述するような機能を位置情報フレームワークに付けるのが良いのではないかと、という流れになるのは当然であり、実際、このサービスを構築する際に、位置情報フレームワークには改良や機能の追加が施され、このサービスに関してのみ言えば、改良・追加された機能を用いて、最小限の位置情報に関する処理のみで良いようになっている。しかしながら、この改良・追加された機能は、このサービス専用の機能という感じが否めなく、また、このような特定のサービス専用のような機能をつけることは、位置情報フレームワーク自体の煩雑化や性能に影響してしまい、結局は、多くの人が気軽に位置情報サービスを提供する土台としてふさわしくないものになってしまうのではないかと、いうことを強く感じた。むしろ、このサービスを構築する際には、本論文で提案するフレームワークを使用しない方が、すっきりとしたソースコードになるのではないかと試したところ、位置情報の変換等に関する部分は、多少複雑になってしまったものの、全体としては、非常に分かりやすく、フレームワークを使用した場合の80%程度の行数で同程度の機能を提供するサービスを構築することができた。このようなことから、やはり、本論分で提案しているフレームワークは、あくまでもサービスにとって有用な位置情報が領域という概念によって表現できる場合にのみ有用なサービスであると言えると思う。

ただ、このサービスにおいて、この提案するフレームワークが完全に不要かというとはそうではなく、ユーザーとの会話をする部分の処理は、元々、このフレームワーク上に、サービス固有のデータをサービスサーバー上と携帯端末上の間でやり取りする機能が備わっているため、非常に少ないわずかな数行程度のコードで実現することができた。よって、位置情報だけでなく、サービス固有のデータを多数やり取りするような場合には、本論分で提案するフレームワークを使用することも検討に値すると思われる

(4) 日常一般に利用されるナビサービス

4章ではこれまで本論文で提案するフレームワークの有用性について、実際の位置情報サービスを通して検証してきたが、ここでは、各種エミュレーターを用いて、近い将来の技術によって可能になるであろう新しいナビゲーションサービスについて実験し、その有用性を検証していきたい。

なお、ここで言うナビゲーションサービスとは、現在一般的に利用されている、迷った時などに利用する、目的地までの経路を表示するツールではなく、日常一般に利用する通学路や通勤路などの経路上において、最適な速度は一体どれくらいであるのかリアルタイムで計算し、その結果を用いてユーザーに指示を出すものである。具体的には、以下のようなサービスである。

- 基本は「次の信号が赤信号なら急いでいく必要はない」という考え方
- 利用する経路とその経路上での予想される平均速度を予め設定しておく
- その上で、ユーザーの現在位置・速度と信号機や乗り物の状況から最適な速度を計算する
- そして、その計算された速度に従って、ユーザーに指示を出す

- 予め設定する経路には、自宅や駐輪場など自分にとって意味のある場所を追加可能
- 追加された場所で費やす時間などもきちんと考慮してナビゲーションをする

つまり、現在の速度のまま移動した場合は、赤信号での待ち時間が多いと予想されるが、もし、平均速度より20%速い速度で経路全体を移動した場合、待ち時間が非常に少なくなると予想されるのであれば、もう少し速く移動するように指示を出す、というようなナビゲーションサービスである。

実際に、GPS エミュレーター・社会情報エミュレーターを用いて実験をしている様子を図4.6に示す



図 4.6 ナビゲーションサービスの実験の様子

このナビゲーションサービスにおいて、最も問題になるのは、どのようにしてユーザーの最適な速度を決定するか、というそのアルゴリズムである。今回は、最もストレスなく、設定した経路を移動できるようにする、ということを実現するために、社会情報エミュレーターから得られた情報を用いて、経路上で設定された平均速度を-50%~+100%と変化させた場合に、待ち時間0で最も遠くまで行くことができる場合を最適な場合とし、その最適な場合に基づいて、現在のユーザーが取るべき最適な速度を算出している。この計算方法は、必ずしも正しいとは言えないが、実際に実験を重ねた結果、GPSの測位間隔が短く、ある程度の測位精度があれば、十分有用であることが確認できている。ただし、この「GPSの測位間隔が短く」というのが非常にやっかいな問題で、当初、数秒程度の遅延であれば、特に問題はないと思われていたが、実験を重ねる中で、この数秒が非常に大きな問題を含んでいることが分かった。考えてみれば当たり前のことではあるが、ある時刻 t_1 までに送られてきた位置情報を元に、サービス側は最適な速度を計算し、その結果を時刻 t_2 に返すわけだが、時刻 t_2 では、ユーザーは既に移動しており、時刻 t_1 の時点までの情報で算出された最適な速度は、時刻 t_2 の時点での状況においては、必ずしも最適な速度ではないのである。実際、このアルゴリズムを用いると、サービス側では、時刻 t_1 から時刻 t_2 までの間、時刻 t_1 の時点までの情報で求めた最適な速度に従ってユーザーが移動することを望むのに対し、ユーザーは、その最適な速度を時刻 t_2 まで知り得ることができないので、最適でない速度で移動せざるを得ない。こうして、サービス側とユーザーとの間で、常にギャップが生じるこの計算方法を用いて計算される最適な速度は、

ある速度を平均として、波のように速くなったり遅くなったりすることを繰り返してしまう。しかしながら、この時刻 t_1 と時刻 t_2 の間隔が短くなればなるほど、この波の周期は大きくなる。よって、ユーザーへの速度の指示に、ある程度の幅を持たせるようにしてやると、常にユーザーへ最適な速度を指示してやることも可能になってくる。つまり、サービスサーバーが社会情報エミュレーターから得ているような、信号機のタイミング情報や乗り物の状況といった情報を誰でもが入手することができ、携帯電話に搭載されるGPSの測位間隔・精度がある程度良いものになれば、日常一般に利用する経路を最も効率良く移動するためにはどうしたら良いのか、といったことを指示してくれるナビゲーションサービスは十分に実現可能であると思われる。

5. 考察及び今後の展開

本論文における最大の問題点は、やはり、提案した位置情報フレームワーク上で位置情報を直接扱うような位置情報サービスを構築しようとすると、どうしても煩雑になりすぎてしまい、フレームワークを利用するメリットが少なく、また、誰でも気軽に位置情報を利用したサービスが構築できる、という目標を達成できなくなってしまう点であると思う。ある見方をすればこれは当然である。なぜならば、元々最初の設計段階において、その位置情報サービスにとって有用な位置情報を領域という概念で表される情報に変換することによって、位置情報に関する様々な知識や複雑な処理からサービス提供者を解放し、誰でも気軽に位置情報サービスを構築できるようにする、ということを基本にしているため、その領域という概念で表すことのできない位置情報サービスは、このフレームワークとは相容れないわけである。しかしながらこのことは、もう1つの可能性を示唆している。つまり、全ての位置情報サービスを誰もが気軽に作成できる、ということは不可能なのではないか、ということである。どのような方法で解決策をとったとしても、結局位置情報を扱わなければならないのであれば、むしろそのようなサービスは、位置情報に関するきちんとした知識を持つプロフェッショナルにまかせてしまう、と割り切ってしまう方が良いのではないだろうか。このような考え方をする場合、本論文で提案するフレームワークは、限定されたサービスにのみに対応するものの、それほど大きな問題点もなく、誰でもが気軽に位置情報サービスを提供できる、という一つの大きな目標を達成できているのではないだろうか。

しかしながら、この問題点が解決されれば、さらに汎用的に使用することのできる位置情報サービスのフレームワークとなることは確実であり、この問題点の解決方法についても少し考えてみたいと思う。まず、この問題点を解決する最大のポイントは、位置情報の変換である。領域という概念を持ち出し、その領域という概念に位置情報を変換しきれなかったためにこの問題が起こるのであるから、領域に変わる、位置情報をうまく隠蔽し、重要な情報のみ抽出できるような新たな概念を導入し、その新たな概念に従ってフレームワークを構築すれば、この問題点は、一気に解決できてしまう可能性は十分にあり、この点についての研究をしてみるのも大変興味深いテーマである。

もう1つの大きな問題点としては、本論文では、エミュレーター等を通して、理論上、有効な位置情報サービスが構築可能なことは十分に検証できたが、果たして、

実際にそのサービスが現実可能であるかどうか、全く触れていない点にある。つまり、本論文では、暗黙のうちにいくつかの仮定をしている。それは、例えば、データ通信料についてはいくら費用がかかってもかまわず、どれだけネットワーク帯域を使用しても良い、といった仮定や、位置情報サーバー・サービスサーバーは誰かがきちんと管理をしている、といったような仮定である。これらの点は、実際に、サービスを現実させようとしたときに、自力では解決することのできない問題点となることが十分予想されるが、実は、これらの問題は、1つだけ大きな解決方法が存在する。それは、この位置情報フレームワークごと、携帯端末上に搭載してしまうのである。つまり、現在は、携帯端末をデータを表示するためのクライアントとしてしか使用していないが、もし携帯端末の処理能力やメモリなどの性能が飛躍的に向上した場合、何も携帯端末をデータの表示専用にせずに、携帯端末内でGPSから得られた情報を処理し、その処理され

た位置情報を領域情報へと変換することも可能であるし、携帯端末から、ネットワークを通じて、社会的な情報を取得することも可能であるし、それらの情報から位置情報サービスを構築する、ということも十分に可能なわけである。このような状況が現実的になったときこそ、真に、誰でもが気軽に日常一般に利用する位置情報サービスを構築することに対しての価値が出てくるのではないだろうか。

6．参加企業及び機関

プロジェクト管理組織

株式会社オープンテクノロジーズ

7．参考文献

なし