GPS軌跡を用いた集合知による道路地図作成システムのためのRaspberry PiとArduinoによる可搬式信号機の設計

佐藤 永欣[1,a)]

概要：カーナビゲーションに使用されている地図は、日本国内ではここ数年、1ヶ月に1度程度の頻度で更新されるようになった一方、地図データが整備されていない発展途上国などでは、そもそもナビゲーション自体が望めないという問題がある。近年、Open Street Map のようなオープンな地図データも普及し出している。国の機関による基本的な測地データが使用可能な国ではこれらの測地データに基づいているものの、データの更新はボランティアベースであり、どうしても更新は遅くなる。そこで我々は、車両の走行軌跡に基づく簡易な測量により道路地図を作成するシステムを開発している。道路地図作成にあたっては、交差点の検出、および交差点に付属する信号機の有無の検出が必要であるが、従来我々が実験に用いていた信号機は可搬性に乏しかった。今回、Raspberry Pi と Arduino を用いて、可搬性に富み、比較的自由な配置を可能とした実験用信号機を開発したので報告する。

キーワード：集合知, GPS, 軌跡, 交差点, 信号機位置, 推定, Raspberry Pi, Arduino

A Design of Portable Traffic Signal for a Raid Map Survey System using GPS Trajectories as Collective Inteligence

Abstract: Maps equipped in car navigation systems are now updated in monthly in Japan, there is still problems that there is no govermental official detailed maps in such as developping countris. In these some years, open data maps such as OpenStreet Maps are now process to diffusion. In these kind of maps, mainly land surveying data by govermental organization is used, however, updating open data maps is not so fast since it is done in volunteer based. Therefore, a rapid map survey system has been developped. Detecting a crossing and its signals are important to create data for road map. However, our previously used traffic signal for experimentation is not enough portable. In this paper, it is reported that new traffic signal with high portability implemented using Raspberry Pi and Arduino.

1. はじめに

現在普及しているカーナビの地図は、新道路開通後に測量、地図会社による地図データ作成、カーナビメーカーによるカーナビの機種ごとへの対応などの作業を経てユーザに配布され、更新されている。地図会社による測量と地図更新のサイクルは1、2ヶ月である。この際、地図会社は自治体などから新規に開通した道路の情報を入手し、測量すべき場所を決めているが、これは国道や県道などの幹線道路に限られる。このため、住宅地などの市町村道では1、2年の間更新されないこともある。また、ユーザへの配布はネットワークによるもの、DVD-ROM 等の媒体によるものが一般的である。前者の場合で1、2ヶ月に1度、後者の場合で年に1度地図が更新されるのが一般的である。このように、地図全体の迅速な更新という面で問題がある。

そこで我々は、カーナビが搭載している衛星測位システムによる軌跡を多数収集し、これらを集合知として用いることで道路位置を推定するシステムを開発している。衛星測位システムとしては GPS が広く知られており、もっとも利用されているが、近年のスマートフォンに搭載された衛星測位システムは Glonass に対応している。Glonass やその他のシステムを含んだ衛星測位システム全体が一般にまとめて GPS と呼ばれているのが現状である。

通常、カーナビは測位誤差に対応するため、マップマッチングを行い最寄りの道路などに現在位置を補正してい

1 岩手県立大学
Iwate Prefectural University, Takizawa, Iwate, 020–0193, Japan
a) nobu-s@iwate-pu.ac.jp

ⓒ2014 Information Processing Society of Japan

る。しかし、新規に開通した道路や地図に記載されていない道路では、現在位置が頻繁に変わるなどマップマッチングに失敗する。我々のシステムでは、これをトリガとしてカーナビが軌跡データをサーバに送信することで多数の衛星測位システムによる軌跡データを収集し、道路位置を推定する。

一般的に用いられている GPS 受信機の測位の精度は高々数 m 程度であり、中には数十 m の誤差を含むこともある。Glonass のみによった場合、さらに精度が悪い。しかし、収集した軌跡データを統計処理する際にロバストな方法を用いること、可能なかぎり多数の軌跡データをも用いることで誤差の影響を最小化し、数メートル程度の精度で道路位置を推定できる [1][2][3]。自動車を運転中のドライバーの興味関心は、道路の大まかな形や、交差点などにおける道路の接続の情報であり道路の正確な位置ではないため、カーナビ用の地図としては数メートル程度の精度で十分である。カーナビの地図における Point of Interest (POI) で、重要なものに交差点と信号機がある。信号機は自動車を運転する際に目印などとしても使用されている。信号機が設置されている場所は交差点、横断歩道、長大トンネルの入口などである。このうち、POI として重要なのは交差点に設置されている信号機と横断歩道に設置されている信号機である。通常、交差点の信号機は時間による制御、横断歩道の信号機は押しボタン式である。

一般に、交通関連の研究ではシミュレータが用いられることが多い。これは車両の位置に誤差が含まれないことを前提としているためである。本研究では、大きな外れ値がしばしば含まれる誤差のある車両位置のデータを元に道路地図を作成することを目的としているため、実車による実験が欠かせない。このため、大学構内の道路を等を利用し実験を行ってきた。この実験に必要となる信号機も製作し仮設して使用してきたが、1 個の制御ユニットで交差点に 4 台設置する信号機の電球の ON/OFF を集中制御で切替えており、有線での接続が必要であったため、設置場所を柔軟に選べないなどの問題があった。そこで、信号機全体の制御ユニットから電球の ON/OFF 制御を切り離し、1 台の信号機の電球の ON/OFF を行う制御ユニットを新たにもうけることとした。また、複数個設置する交差点の信号機全体の制御ユニットと、各信号機の制御ユニット間を無線接続とし、各信号機への配線は電源のみとした。本論文では、この信号機の設計と実装について述べる。

2. 関連研究

GPS (Global Positioning System) は上空にある衛星からの信号 (時刻データ、衛星軌道情報) を GPS レシーバで受け取り、衛星の位置をもとにして現在位置を知る衛星測位システムである。同様なシステムとして旧ソ連で開発され、信号の変調方式やフォーマットなどを GPS に合わせてシステムが再構築された Glonass にも対応している受信機が最近増えている。これらの受信機では、Glonass を含めた衛星測位システム全体が GPS と呼ばれていることが多い。これは元もとあった GPS 受信機に Glonass や他の衛星測位システムの受信機能が追加されたためである。

現在のカーナビは地図付の衛星測位のためのレシーバであり、内蔵されている衛星測位モジュールによって現在位置の情報を計算し、現在位置を地図上に表示している。また、スマートフォンやタブレット端末がカーナビの代わりとして使用される場面も増えている。しかし、衛星測位システムによって得られる現在位置の情報には、レシーバの性能差や測位に用いる衛星の配置、天気、気温などの影響により多少の誤差が含まれている。その情報だけを用いて現在位置をカーナビに表示すると走行中の道路とは全く関係の無い場所を現在位置として表示してしまうことがある。

この問題を解決するために、マップマッチングを行う。マップマッチングは、衛星測位システムによって得られた現在位置を、カーナビ内に含まれる地図情報を用いて道路上になるように補正する処理のことである。これによって、現在位置をかなり正確に知ることができる。しかし、マップマッチングの処理には次のような問題点がある。

- 道路構造が複雑化したため、マップマッチングによって違う位置を現在位置として補正してしまう。例として、隣接・並行している一般道路と高速道路を混同する場合などがある。
- カーナビ内の地図に反映されていない新道路を走行すると、マップマッチングの影響で明らかに異常な位置に補正してしまう。

しかし、衛星測位システムの測位精度は建物などの影響を受けてかなり変動するため、マップマッチングは必須である。また、カーナビはマップマッチングのほかにもルート案内など、その機能の多くを地図に依存している。このため、カーナビの地図は、新道路の開通後すぐに (可能であれば開通と同時に) 更新されるのが望ましい。

新道路が完成してから地図に反映されるには時間がかかる。日本国内の場合、新道路が開通すると、地図会社が位置や形状などを測量する。その後、地図データの編集を経てカーナビ用地図が完成するため、どんなにはやくても 1 週間ほどかかってしまう。近年は、道路を作る行政組織が地図会社に、新しい道路の開通情報や図面を提供するなど、できるだけ早く完成した道路を地図に反映させようとしている。しかし、住宅地などの市町村道ではこのような情報が無いため、1、2 年の間更新されないこともある。発展途上国では、地図そのものが未整備であったり、政府機関による定期的な測量が行われていないなど、そもそもカーナビ用地図を提供するためのインフラが揃っていないこともある。

衛星測位システムによる走行軌跡のみから地図を自動的

ⓒ2014 Information Processing Society of Japan

に更新しようとするシステムは、我々が知る限り存在していない。一方、GPS による走行軌跡を集合知として用いて、渋滞や道路が通行可能であるかどうかの情報を自動的に収集し、ユーザに提供するシステムは既に実用化されている。このようなシステムとして、インターナビ・フローティングカーシステム [4]、自動車通行実績情報マップ [5]、スマートループ [6] などがある。また、路側に設置したプローブ用機器からの情報といった、GPS 以外の情報による道路交通情報システムとして VICS[7] がある。

自動車通行実績情報マップは、2011 年 3 月 11 日に発生した東日本大震災で被害を受けた地域を中心として、自動車の走行実績があった道路を地図上にプロットし提示するサービスであり、震災から 3 日後の 3 月 14 日から提供を開始している。このサービスはカーナビメーカー各社（HONDA、パイオニア等）のカーナビを搭載した車両の走行実績データや、国土地理院が公開した東北地溝道路規制情報を取集し、Google Map 上にプロットして公開している。

これらは、地図上の情報の自動的な更新を実現しているものの、更新の対象となっているのは、渋滞情報、通行止め情報といった交通情報であり、動的に変化することがあらかじめ想定されている情報である。すなわち、既に存在し、地図に掲載されている道路に、動的に変化する情報を付加しているだけである。また、VICS では道路の区間ごとに ID を振って渋滞情報を提供するため、カーナビの地図があらかじめそれらの情報を持っている必要がある。

OpenStreetMap は、オープンに利用できる地図をユーザが GPS 受信機等の機器を利用して作成しようというプロジェクトである [8]。各地の住人が居住地の周囲を GPS 機器などによって測位してまわり、そのデータに基づいて地図が作成され、公開されている。この際、海岸線や主要な道路などのデータは、国土地理院のような政府機関が (パブリックドメインやそれに近い形で) 公開しているデータを用いることもある。いずれにせよ、自動的な地図の作成は考慮されていない。

3. 集合知による道路位置推定手法

我々が以前提案した道路位置推定手法 [1][2][3] は、新しく道路が開通した際に迅速に道路の形状と位置を推定し、カーナビに提供することを目的としている。したがって、測地の分野で求められるような数センチメートル以下の精度での地図作製を目的とはしていない。ここでは、その概要について述べる。提案システムの概要を**図 1** に示す。

まず、準備段階として、緯度経度双方を 0.1 秒刻みのメッシュとした平面空間を用意する。メッシュの各正方形をグリッドと呼ぶことにする。なお、緯度 0.1 秒の長さは地球上のどこでも同じであるが、経度 0.1 秒の長さは緯度によって異なることに留意する必要がある。すなわち、高緯

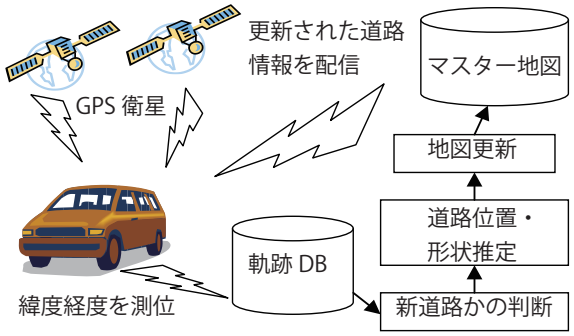

図 1 提案システムの概要
Fig. 1 Overview of a proposing system

度地方では経度 0.1 秒の長さが赤道付近に比べて短くなる。この点は。本システムを高緯度地域で使用する場合に問題となる可能性がある。本システムでは、次のように道路の位置を推定する。

(1) マップマッチングできなかった走行軌跡を収集する
(2) 走行軌跡の走行方向を分類する。通常は往復 2 方向分、一方通行でも一方向分となる
(3) 方方向への走行軌跡について、グリッド内を通過した数をカウントする
(4) 次のように通過カウント数がもっとも多いグリッドに道路があると推定する。すなわち、最頻値により道路位置を推定する。これは測位結果の外れ値の影響を避けるためである
  (a) 現在注目しているグリッドの隣接グリッドから、現在注目しているグリッドを通る軌跡の走行方向が最多である方向のグリッドをひとつ選択する。このグリッドを、仮に次のグリッドとする。
  (b) 仮の次のグリッドの両隣を探索する。ここで両隣とは、現在注目しているグリッドにおける走行軌跡の最多の走行方向に垂直な向きに隣接する 2 グリッドである。
  (c) これらの計 3 グリッドの中から、通過する走行軌跡の本数が最多であるグリッドを次のグリッドとして選択する。4a にもどる。
(5) 反対方向への軌跡が存在する場合、4 を繰り返す。これで上下 2 車線分の道路位置が推定できる。
(6) 上下 2 車線分の軌跡を次の処理を繰り返すことにより統合する。
  (a) 任意の方方向のグリッド列を選択し、それに垂直な方向にもう一つのグリッド列を探す
  (b) 二つの対応するグリッド列の中間となるグリッドを道路の中央とする

このようにして統合されたグリッドには幅情報がない。

また、グリッドがジャンプしている場合は補間する

このようにして推定された道路位置の例を**図 2** に示す。こ

ⓒ2014 Information Processing Society of Japan

図 2　学内周回道路の位置推定結果
Fig. 2　Estimated Position of Runiversity's Road

の他、信号機の位置の推定を実現している。信号機位置の推定は、交差する軌跡、合流する軌跡などにより交差点の検出を行った後、青信号を通過する軌跡であれば速度はほぼ変化せずに通過すること、赤信号であれば交差点の手前で車列を作った状態で停止することを利用している。ただし、前方の車両の右折待ちなどにより、交差点進入後に停止した場合の処理方法など、一部で検討が必要である。また、交差点でないところに歩行者用押しボタン信号機のみが設置されているような場合なども、検出の自動化のためには十分な検討が必要である。

4.　信号機の設計

まず、以前から使用している実験用信号機の外観を**図 3** に示す。今回はこの信号機の構造部分等を再利用し、制御方法を変更する。以前の制御部分の仕様は、Windows PC に USB 接続された USB デジタル IO (Lifetron DO-16(U)) の 16 ビット分の出力を使用して、4 台の信号機の点滅を全て制御していた。このため、信号機までの配線が多く、また長さにも制約が生じやすいため、自由に交差点を構成することができなかった。電球の点滅の制御自体は、秋月電子の Solid State Relay (SSR) キット (35A タイプ、K-00210、5V TTL 入力、正論理、フォトトライアック絶縁) を使用している。しかし、SSR を 1 ヶ所に集中して配置せざるを得なかったため、信号機 1 台につき AC 100V の配線が 3 本必要となってしまい、設置の自由度を著しく制限していた。

今回、Windows PC と USB デジタル IO をそれぞれ Raspberry Pi と Arduino に置き換え、両者の間を無線 LAN で接続することにより、柔軟な配置を実現する。具体的には、WiFi アダプタを装着した Raspberry Pi でマスターとなる制御プログラムを動作させ、各信号機に設置した Wi-Fi シールドを取り付けた Arduino に信号機の点灯状態を変更するコマンドを送信することにする。Arduino のデジタル出力を SSR への入力とし、信号機の点灯を行う。これに伴い、1 台の信号機の下に集中的に配置されていた SSR を各信号機の下に設置する。これにより、無線 LAN が届く範囲であれば、AC 電源の確保という問題が残るものの、柔軟な信号機の配置が可能である。これは、数十メートルの片側交互通行の実験のようなことを可能にする。

具体的な構成は以下の通りである。

Raspberry Pi　マスターコントローラとして動作する。FreeBSD-10.0-RELEASE をインストールし、無線 LAN アクセスポイントとして使用する。

- 無線 LAN アダプタは Buffalo WLI-UC-GNM2 を使用した。run(8) として認識される
- hostapd を使用して無線 LAN アクセスポイントとした。Arduino の CC3000 WiFi ボードの標準的な動作に合わせて DHCP で IPv4 アドレスを配布する
- ruby で記述された信号制御マスターが稼働する

Arduino　SparkFun の CC3000 WiFi Shield を取り付け、スレーブとして動作させる。

- CC3000 が使用しないピンに信号機の点滅を制御するための SSR を取り付ける。SSR に使用されているフォトトライアックはロットによって異なるようであるがフェアチャイルドの MOC3041 であった。入力は 50mA 程度流すことが想定されている。順方向

図 3　実験用信号機
Fig. 3　A Traffic Signal for Experimentation

ⓒ2014 Information Processing Society of Japan

ギャップ電圧は 1.2V (typ.) である。このため Arduino で十分ドライブ可能である

- CC3000WiFi ボードとの通信と信号機の制御に使用していない Arduino の IO ピンは、信号機 ID 番号を設定するために使用される
- Arduino は上記の Raspberry Pi 上で動作する信号機制御マスターに TCP 接続し、自分の信号機 ID 番号を申告した後、コマンドを待ち受ける

図 4 に今回 Raspberry Pi と Arduino により改造した信号機の構成を示す。Raspberry Pi は点線で囲まれた部分が信号機 1 台分であり、今回は 4 セットある。必要に応じて増減できる。

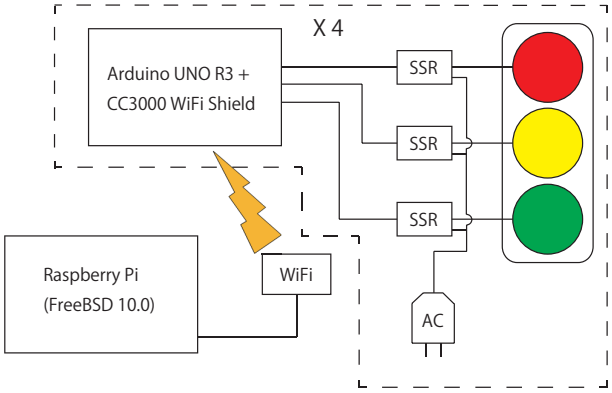

図 4　無線化された信号機
Fig. 4　WiFi Controled Traffic Signals

信号機の制御は 2 モードある。すなわち、通常の青、黄、赤を繰り返すモードと、赤点滅、黄点滅のモードである。信号機の制御は単純なマスター・スレーブモデルを採用している。すなわち、マスターである Raspberry Pi で実行されている信号制御サーバからコマンドを送信すれば、即時それが反映される。

信号機を制御する Arduino のプログラムが受け付けるコマンドは次の通りである。

G　青色を点灯する
Y　黄色を点灯する
R　赤色を点灯する
O　全て消灯する
BY　黄色を 2 秒周期で点滅する
BR　赤色を 2 秒周期で点滅する

このうち、BY と BR は Arduino の内蔵タイムカウンタを使用するため、時間の経過により、同期がずれることが予想される。このため、Arduino 側の制御プログラムとして実装されているのみであり、通常の交差点を作成する際には使用しない予定である。すなわち、赤点滅、黄点滅のモードでは、Raspberry Pi で動くサーバから、Y または R と O コマンドを交互に送り続ける。

なお、Arduino の CC3000 ライブラリの都合上、行単位の IO も実装可能であるが、ライブラリとして実装されているわけではなく、メモリ容量の余裕の問題もあるため、実質上 1 文字ずつの I/O しかできない。ライブラリで実装されているのは 1 文字ずつの I/O と、データが到着しているかの peek のみである。このため、Arduino の信号制御プログラムが受け取るコマンドは、1 文字単位で条件分けできるようにしてある。例えば、この例では、Arduino の信号制御プログラムは B の後は Y または R がただちに読みだし可能なはずであり、読みだし不可能、または Y または R 以外の文字が来ることはありえない、というロジックで実装されている。理解できないコマンドが送られた場合は単純に無視する。

各信号機は信号機 ID を持っている。信号機 ID は、Arduino の信号機の制御と CC3000 Wi-Fi ボードの制御に使用されないピンを使用して、ディップスイッチまたはジャンパピンで設定する。少なくとも 3 本のピンが信号機の制御と CC3000 Wi-Fi ボードの制御以外の目的に使用可能である。

交差点を構成する信号機は、次のような階層構造をもっている。この階層構造にしたがってクラスが設計されている。

信号機セット　組になって制御される信号機のセット。このセットに含まれる信号機は同時に青になる。たとえば、十字路を構成する道路の二方向に向けて設置される信号機である。各信号機セットは開通、非開通の二つの状態を持つ。開通状態では、各信号機セットはそれぞれの作動時間を持ち、作動時間終了後に非開通状態に遷移する。この際、あらかじめ設定された次に開通させるべき信号機を開通状態に遷移させる。非開通状態では、他の信号機の開通が終了し、自分の順番が来るまで待機する。順番が来たら開通状態に遷移する。

信号機　単体の信号機。信号機はランプを含んでいる。赤以外のランプは信号機セットの開通状態と対応しており、赤のランプは開通状態と対応している。赤以外のランプはそれぞれの点灯時間を持っている。信号機セットを構成する信号機がすべて非開通状態、すなわち、赤に遷移したら、信号機セット全体が非開通状態に遷移する。

ランプ　信号機の赤、黄、緑のランプ。このほかオプションんで右折矢印を設置可能である。黄と緑、右折矢印にはそれぞれの点灯時間がある。これらは各個に設定可能である。したがって、右折矢印や、時差式信号機の設置も可能である。

これらの信号機セット、および信号機は、RaspberryPi 上で動作する ruby で記述された信号機制御用サーバにより制御される。

上記の交差点を構成する各信号機セットは次のようにして制御される。

ⓒ2014 Information Processing Society of Japan

(1) 交差点を構成する各信号機セットを初期化する。信号機制御用サーバが起動したあと、各信号機の電源を入れ、Arduino 上で動作する信号機制御プログラムが、各自の信号機 ID をサーバに申告する。その後、信号機はサーバから赤を点灯するコマンドを受信し、赤を点灯させる
(2) 交差点を構成するすべての信号機が接続したら、信号機制御用サーバは交差点の制御を開始する。あらかじめ設定された開通させる信号機セットのうち、一番最初の信号機セットを開通させる。この際、各信号機に緑ランプを点灯するコマンドが送信される。
(3) 信号機制御用サーバは、開通させた信号機セットが時間の経過により非開通状態に遷移したら、次の信号機セットを開通状態に遷移させ、その信号機セットを構成する信号機に対して緑を点灯させるコマンドを送る。このステップを交差点が不要になるまで繰り返す

交差点の動作モードは上記の通常モードの他に、点滅信号モードがある。点滅信号モードは現在のところは実装されていないが、黄または赤の点滅を継続するモードである。通常モードと点滅信号モードは相互に切り替え可能である。通常モードから点滅信号モードへの移行は、信号機セットのうちの一つが、開通状態から非開通状態に遷移したタイミングに合わせて実行される。

5. 今後の実験計画

この信号機の (再) 実装により、模擬交差点への柔軟な信号機配置が可能となったほか、無線 LAN が届く範囲であれば、片側交互通行の実施が可能である。この際、図 2 に示す学内の周回道路を占有して交差点や片側交互通行を模擬し、スマートフォンなどの GPS 受信機を搭載した機器を車載し、軌跡データを収集する。

交差点に存在する信号機の検出が可能であることはすでに [3] で示した。十字路を構成して先行車の右折待ち、および信号待ちの検出が可能なことは示したが、用意できた車両台数の問題から、一方向からの実験しか行っていない。このため、二方向、四方向からのトラフィックが存在する環境下で実験を行う必要がある。また、右折レーン及び右折矢印信号などの検出が可能であるかどうかも実験を行うべきである。

また、信号機制御モジュールと信号機の設置の自由度が上がったため、災害時などにおける片側交互通行などの状況を大学内の周回道路を使用して実現できる。Wi-Fi のみによる通信距離自体は Arduino WiFi Shield と Rasperry Pi に装着する無線 LAN 機器に依存するが、これらを直接接続する必要は無く、間に 3G データ回線等をはさむことも可能である。

片側交互通行を行う実験では、GPS 受信機を搭載した車両を多数用意し、信号待ちの渋滞を検出する。そのうえで、片側交互通行区間の長さ、待ち時間、交通量の片側交互通行を検出できる下限を探りたいと考えている。さらに本システムにデータを提供できる車両の交通量全体に対する割合を評価することも重要である。図 2 に示す学内の周回道路を完全に占有して、制御された環境を作る必要があるかもしれない。

6. まとめと今後の課題

本稿では、航法用 GPS 受信機による数メートルの誤差を含む軌跡データから地図を自動作成するためのシステムにおいて、交差点における信号機の存在を検出する実験を行うために必要な、信号機の設計と実装について述べた。すでに実験を行うための信号機は存在していたものの、Windows PC に USB 接続されたデジタル IO デバイスを用いて制御していたため、接地の自由度が低かった。また、32bit Windows 用のドライバしか供給されていなかったため、今回、Raspberry Pi と Arduino を用いて再実装した。これに合わせて、各信号機を制御する Arduino によるモジュールと交差点全体を制御する Raspberry Pi の間を Wi-Fi 接続することで、設置の自由度を大幅に高めた。接続は Wi-Fi による直接の接続のみではなく、IP レベルで通信が可能であればよいため、Arduino 用 CC3000 WiFi ボードを適切ななルーティングを行うアクセスポイントなどに接続するように設定すれば、数キロメートルにわたる片側交互通行の実施も可能である。

参考文献

[1] Hiromitsu Sugawara, Nobuyoshi Sato, Tsuyoshi Takayama, Yoshitoshi Murata, Early Evaluation of Road Width Estimation on Rapid Road Map Survey System Using GPS Trajectories as Collective Intelligence, The 3rd International Workshop on Information Technology for Innovative Services (ITIS 2011), in conjunction with The 14th International Conference on Network-Based Information Systems (NBiS 2011), pp.547-552, Tirana, Albania. (2011.9)

[2] 菅原弘光, 佐藤永欣, 高山 毅, 村田嘉利, 集合知としての走行軌跡による道路の車線数推定手法の評価, マルチメディア通信と分散処理ワークショップ, pp.154-159. (2011.10)

[3] 佐藤 永欣, 高山 毅, 村田 嘉利, GPS 軌跡を用いた集合知による道路地図作成システムにおける信号機位置推定法, マルチメディア通信と分散処理ワークショップ, pp.242-248. (2012.10)

[4] ホンダ技研工業株式会社, “internavi”, http://www.honda.co.jp/internavi/

[5] Google, 東日本大震災 - 自動車通行実績情報マップ, http://www.google.org/crisisresponse/kiroku311/chapter_12.pdf

[6] パイオニア株式会社, スマートループ, https://www.smartloop.jp/smartloop/index.html

[7] 財団法人道路交通情報通信システムセンター, VICS, http://www.vics.or.jp/

[8] OpenStreetMap, http://www.openstreetmap.org/

ⓒ2014 Information Processing Society of Japan