AN EXTENSIBLE AND INTERACTIVE SOFTWARE AGENT FOR MOBILE DEVICES BASED ON GPS DATA

Francesco Fornasari, Claudio Montanari
Software Engineer

Barbara Furletti
IMT Institute for Advanced Studies Lucca
Piazza S. Ponziano, 6 – 55100 Lucca, Italy

ABSTRACT

This paper presents a real-time software agent for monitoring and supporting users that move along predefined paths. We investigated appropriate models for geographical data and communications (based on cellular networks), also an *ad-hoc* localization algorithm based on GPS technologies has been implemented. We developed an Interactive Road System by installing the whole application on a mobile device and we customized it for supporting drivers. The power of the system is based on its flexibility and the possibility to extend the set of action, the agent has to perform.

KEYWORDS

GPS, GNSS, Localization System.

1. INTRODUCTION

In the last two decades we witness an increasing trend in the use of positioning and navigation systems in several domains including: automated car navigation, emergency assistance, fleet management, person/asset tracking, collision avoidance, environment monitoring, and automotive assistance. Because of the relevance of spatial information and localization in the common use, the market boosted the research for the development of more and more sophisticated systems able to interact with users and provide information. Nowadays, the Global Positioning System (GPS) is the most popular localization system used in commercial software applications, and installed on common devices. It provides reliable positioning, navigation, and timing services to users on a continuous worldwide basis. A GPS receiver acquires signals from at least four satellites and calculates a three-dimensional position [2, 3]. Basing us on this main technology, we studied a **geographic model** and an interactive GPS system, applicable in all situation in which we need to provide the final user of information, not only related to geographical map events (such as, point of interests). We also provided the system with a **communication model** to increase its interactive ability and for improving the management of the collected data. This last model (supported by GSM/GPRS[1] technologies) connects the user (i.e. the mobile device in which the system runs) to a Service Center. All the system has been implemented and integrated on a commercial palm device (a PDA). We customized the tool for a twofold purpose: for monitoring and tracing vehicles that move on predefined roads, and for sending information to the users (i.e. some information about places, urban areas, road paths in which users are located). The paper is organized as follows. Section 2 describes the components of our system. Section 3 supplies some technical details about the implementation and the testing phase. In section 4, the conclusion and some future improvements are discussed.

[1] General Packet Radio Service (GPRS) is a Mobile Data Service available to users of Global System for Mobile Communications (GSM).

2. THE SYSTEM

The system we propose is based on the architecture shown in **Figure 1**. As you can see, the PDA (that is the mobile device with GPS and GSM/GPRS support), plays the key role in the application because it is the main hardware component. Over this platform, a software system has been implemented and installed. The Service Center (SC) represents the remote support for data and information and controls the web-service communication. The satellite is the external component that provides the coordinates for the localization mechanism.

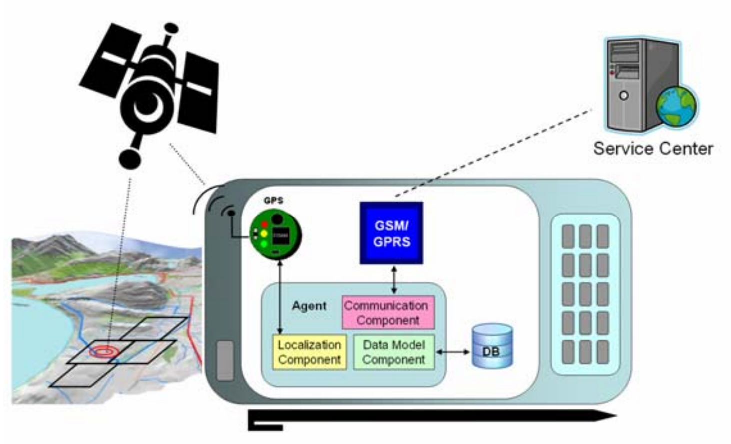

Figure 1. System Architecture.

The agent is a localization software for recognizing geographical areas and for executing related to those areas. It manages GPS and GSM hardware components, and deals with some critical working events (such as the temporary loss of GPS and GSM signals).

In the sections below we explain the software architecture focusing us on: (i) the geographical data model, used for modeling the reality; (ii) the communication model, for data updating and information exchanging between PDA and SC; (iii) the localization algorithm, which represents the core of the agent.

2.1 The Data Model

The data model describes the geographic reality and it is essential for the localization algorithm: the accuracy of the model influences the algorithm results, but as a counter part, the data model has to be as generic as possible for representing a large number of real situations. The territory is divided into sectors (Figure 2a) and each sector has some applicative contexts which contain instructions that a device must execute in particular circumstances. Sectors can cover large surfaces (or the part of them that may be reached by predefined paths), they must not overlap and between them, no holes must be presented. Those facts assure that every device is always into one sector at time.

A sector is identified by the following tuple:

SECTOR (ID, VERSION, GEOPOINTS, VALIDITY, CONTEXT IDs)

The ID is an unique identifier, VERSION and VALIDITY attributes are parameters used for establishing the freshness of geographic information; the VALIDITY is in particular a standard timestamp. The GEOPOINTS data represent geographical points: each point is a couple *(longitude, latitude)*, and each sector is formed by at least three of them. CONTEXT IDs are the unique identifiers of contexts related to the sector. The sector information must exist and must be continually consistent in the system, otherwise the software agent is unable to know the device position. Contexts contain a set of information and in particular the geographical shapes (called geo-objects) and the actions associated to them. It is important to point out that, also contexts can not overlap; in other words, geo-objects can not appear in different contexts, otherwise the agent can not univocally determine where it is located. A context must contain the following information:

CONTEXT (ID, VERSION, VALIDITY, GEO-OBJECTS, ACTIONS)

The ID, VERSION and VALIDITY have the same meaning of those ones specified for the sector. Typical geo-objects can be Virtual Gantry (VG) and Zone. A VG (**Figure** 2b) is composed of two consecutive polygons (such as trapezes), and it is spatially identified by six geographical points (e.g. the vertexes of the two polygons). The VG has three location status: **IN FRONT OF** VG, **BEHIND** VG and **OUTSIDE** VG.

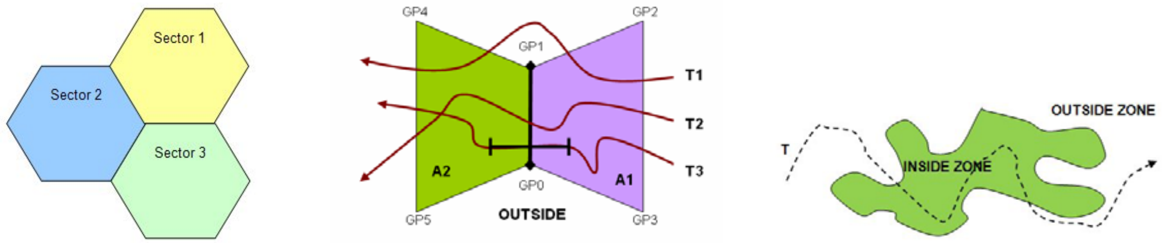

Figure 2. (a) Sectors shape, (b) The Virtual Gantry Geo-Object. (c) The Zone Geo-Object

These status are the only available parameters for understating where the mobile device is actually located on the VG. The agent determines the VG crossing when the VG location state changes from IN FRONT OF to BEHIND without meanwhile reaching the OUTSIDE state. This concept is graphically exemplified in Figure 2b. The six geo-points *GP0, GP1, ..., GP5* delimit the VG formed by the areas *A1* and *A2* that represent the IN FRONT OF and the BEHIND status, respectively.

The three trajectories, *T1*, *T2*, *T3* represent three possible geographical paths followed by a mobile device.

- T1 path: the mobile device is initially located in *A1*; then it moves in an external zone (OUTSIDE state) and, at the end, it reaches the area *A2*. We consider the VG "not-crossed", because the mobile device exits in an OUTSIDE zone after reaching the IN FRONT OF zone. For stating that a VG has been crossed, it is essentially that the mobile device passes through the gantry that is the segment $\overline{GP0\ GP1}$ of our example.
- T2 path: the mobile device passes directly from *A1* to *A2* crossing the segment $\overline{GP0\ GP1}$.The status changes from IN FRONT OF to BEHIND, so we consider the VG crossed.
- T3 path: the mobile device is initially located in *A1*, then it enters in a tunnel where the GPS signal is absent. When the mobile device exits from the tunnel, it is located in *A2*. We consider the VG crossed even if the temporary absence of the signal does not permit rising the instant of the crossing. This rule makes cases of the T2 and T3 paths, similar.

The VG size has been empirically determined during the testing phase. We verified, in fact, that the VG size is strongly dependant to both the mobile device medium speed[2] and the computational power. Obviously, if the VG is small and the mobile device moves at a high speed, the processor is unable to perform in time the calculation related to the localization and the actions. During our experimentations, we learnt that a VG length must be at least three hundred meters for both VG areas, while the VG width have not particular constraints (obviously, the width can not be too short and it must be proportioned to the GPS chipset HDOP[3]).

The Zone geo-object is less complicated than VG. It is, in fact, a closed polygon of at least three sides (in other words, it is composed of a minimum of three geo-points to infinite). When the mobile device enters in the Zone, the status changes from OUTSIDE to INSIDE, while when the mobile device exits, the status changes from INSIDE to OUTSIDE, as shown in Figure 2c.

ACTIONS correspond to the activities that the agent must perform whenever it recognizes a geo-object. The agent can execute single or multiple actions. As you can image, the actions are strongly related to the domain of application, they can be freely defined even keeping into account the computational complexity with respect to the hardware powerful. Examples of actions can be: send a message, send a signal, execute mathematical operations, run an algorithm, display some information, play a sound. The formal representation is:

ACTION (ID, TYPE_OF_ACTION)

[2]For the mobile device medium speed we intend the speed at which the mobile device moves: in our case corresponds to the medium speed of the vehicles on which the mobile devices have been installed.

[3] HDOP: The horizontal dilution of precision allows one to more precisely estimate the accuracy of GPS horizontal (latitude/longitude) position fixes by adjusting the error estimates according to the geometry of the satellites used.

where ID is an unique action identifier and TYPE_OF_ACTION identifies the activity to execute. In paragraph 3, we will show an applicative example of actions. The geographical model just described is able to cover a large number of cases.

2.2 The Communication Model

The Communication Model (CM) describes the communications between the agent and the external environment (such as a Service Center). CM mechanisms are based on *web-services*[4]: a Service Center manages, checks and satisfies the requests sent by mobile devices. The low level communication of a CM is supported by the EGPRS/GSM unit and implemented by the operating system which runs on the mobile device. The CM has two main tasks:

- Updating the data of the mobile device.
- Sending the data collected by the mobile device.

Updates are necessary for avoiding the data inconsistency and obsolescence (this is the case in which geographical data have to be modified or corrected for representing new shapes, or new actions have to be added). This procedure is accomplished by assigning a period of validity to sector and context data. The period has a *renewal date* and an *expiry date* (see Figure 3). In agreement with the standard proposed in [4], the CM can perform the data renewal by using the renewal and expiry dates.

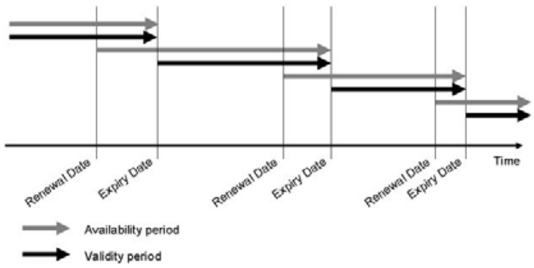

Figure 3. Renewal and Expiry periods.

In general the validity period for sector information and context data may be different. Nevertheless in both cases, the validity period is automatically extended to a new expiry date if no changes are needed. If there is a change, then the new version of the data must be available before the validity start. Data becomes available at the renewal date, and expires at the expiry date.

For putting these specifications in practice, we defined three transactions between SC and a mobile device:

1. Test Transaction (*TT*): TEST_TRANSACTION → RETURNS CODE_OK
2. Sector Updating Transaction (*ST*): SECTOR_UPD(*ID_SECTOR*) → RETURNS SECTOR DATA
3. Context Updating Transaction (*CT*):

CONTEXT_UPD(*ID_CONTEXTS*) → RETURNS CONTEXT DATA.

The transaction *TT* is used for testing the SC availability. The mobile device asks for the presence of the predefined SC by sending a TEST_TRANSACTION and waits for a CODE_OK. If the request expires, the SC is not available and the mobile device can not run properly. In a real scenario, the TT transaction is useful to reduce costs of cellular connections since the amount of bytes exchanged between the agent and the SC is small. The *ST* transaction is used for asking for the sector data. When the agent recognizes the sector, it analyzes the corresponding data. If data are not completed, the agent sends a request to the SC specifying the sector ID and wait for the sector data. The *CT* transaction is used for asking for the context data. The agent analyzes the context associated to the stored ids. If data are not completed (the context has not a version number) or inconsistent (the context version is out of date), it asks for new data by sending the context ids to the SC.

The mobile device must have the possibility of communicate some interesting data to the SC (i.e. data collected during a trip or filtered by some predefined conditions). This data are collected by the agent during the running phase, and are sent to the SC at predefined intervals or whenever a geo-object action orders it.

COMMUNICATE_TO (*MOBILE_ID*, *MESSAGE_DATA*)

[4] It should be useful to have encrypted communications by using W3C XML Encryption standard.

The COMMUNICATE_TO(…) transaction has two fields: the mobile unique identifier and a message which contains the collected data. Examples of data are: the detailed list containing the crossed geo-objects and the GPS time of geo-object crossing event, and the distance between each geo-object. This kind of information is useful when the SC has to check mobile agent trips.

2.3 The Localization Algorithm

The real-time GPS localization is a methodology which permits recognizing the mobile device location on the territory. It uses the hardware receiver configuration and capabilities (in terms of precision and quality) for manipulating NMEA[5] sentences and for extracting geographical coordinates (longitude and latitude), altitude, PDOP and HDOP values and the speed and distance. The agent applies these information to the geographical data stored in the mobile device for understanding if the current position is part of an area. For determining if a point is located inside an area, we used the well known half-line algorithm, based on *Jordan Curve Theorem* [12]. The general idea is the following:

If the half-line hits the area borders an odd number of times, then the point is inside the area. Vice versa, the point is outside the area.

Figure 4. (a) Cartesian Plan with point *A* and five half-lines. (b) Identification of the point Y_C.

With reference to **Error! Reference source not found.**a and **Error! Reference source not found.**b, let:

- H1, H2, H3, H4, H5 be five half-lines,
- *A* be one point drew and located on the Cartesian plane,
- *C* be a counter for keeping track of the number of half-line hits with the polygon borders.

The main steps of the (customized) localization algorithm can be summarized by the following pseudo code:

1. Initialize *C* and set it to *0*.
2. **FOREACH** couple of segment vertexes (V_i, V_j), **DO**:
   a. IF *(NOT(* $V_{ix} < A_x$ *AND* $V_{jx} < A_y$ *)) AND (NOT(* $V_{ix} > A_x$ *AND* $V_{jx} > A_y$ *))* AND *(NOT(* $V_{iy} < A_y$ *AND* $V_{jy} < A_y$ *))* THEN **exec** b.
   b. IF *(* $V_{iy} > A_y$ *AND* $V_{jy} > A_y$ *) AND* **(** *(* $V_{ix} < A_x$ *AND* $V_{jx} > A_x$ *) OR (* $V_{ix} > A_x$ *AND* $V_{jx} < A_x$ *)* **)**
      THEN $C := C + 1$;
      ELSE IF *(* **(** *(* $V_{ix} < A_x$ *AND* $V_{jx} > A_x$*) OR (* $V_{ix} > A_x$ *AND* $V_{jx} < A_x$*)* **)**
      AND **(** *(* $V_{iy} < A_y$ *AND* $V_{jy} > A_y$*) OR (* $V_{iy} > A_y$ *AND* $V_{jy} < A_y$*)* **)** *)*
      THEN IF *(* $\mathbf{Y_c} \geq A_y$ *)*
      THEN $C := C + 1$;
   c. Analyze the next segment.
3. When all the segments have been analyzed:
   IF *C* values is a odd number
   THEN *A* is located inside the polygon
   ELSE *A* is outside.

[5] The **NMEA (National Marine Electronics Association) 0183** Interface Standard defines electrical signal requirements, data transmission protocol and time, and specific sentence formats for a 4800-baud serial data bus. See http://www.nmea.org/pub/0183/ for further details.

The Y_c represents the case of **Error! Reference source not found.**b, and it is calculated by using the expression: $Y_C = Y_L + \frac{Y_R - Y_L}{X_R - X_L} * (X_A - X_L)$. The formula covers the case in which the point is located under an oblique segment. In other words, the oblique segment represents the diagonal of a fictitious rectangle which encloses the point *A*. The algorithm solves the issues related to all polygon shapes with at least three sides and that are not overlapped.

3. IMPLEMENTATION AND TESTS

We implemented a pilot application by using commercial software and hardware as main support basis. We customized the prototype for realizing an ***Interactive Road System*** (IRS). The interactivity has been realized by sending graphical and audio messages to the user that covers a route layered by VGs. The system can help a user by sending graphical and audio messages when he travels on a road where some VGs had been positioned during the geo-graphical data configuration.

As mobile platform we choose the **HP iPAQ 6915**[6] PDA because it offers a complete solution for the connectivity and has a very qualitative GPS support. It is, in fact, a fully mobile phone with EGPRS GSM support and it has the *Sirf Star III GPS* chipset (actually one of the best GPS chip for commercial uses). The processor is an *Intel XScale* 410Mhz with 64Mb of RAM for the programs execution. The operating system is the *Microsoft Windows Mobile 5 Phone Edition*.

We implemented the software in **C#** language by using *Microsoft .NET Compact Framework 2.0* [6]. Those tools allowed us to easily and briefly develop our demonstration environment, without losing performance, and assuring the portability requirement. Moreover, we adopted the *Microsoft SQL Mobile 2005 CE*[7] as database and *ADO.NET* [1] access and data manipulating techniques, and the *Franson GPS Tool* library [2] for the GPS management. For playing some informative text messages we adopted the *Loquendo TextToSpeech* [5] engine.

The SC has been implemented by using the standard J2EE 1.4[8] environment. The SC offers all functionality for configuring sectors, contexts, geo-objects and actions. Moreover, it traces the mobile devices activities for offline analysis of the carried out paths, or online monitoring activities.

The web-services communication allows the interaction between the SC and mobile devices for data downloading. For drawing geo-objects, and geographical data in general, we used the *Google Earth* system support. With the Google Earth software we traced both sectors and geo-objects (like zones) on the earth surface. The sector data have been exported in KML[9] Google format [7] and imported into our SC *Postgres* database.

When the device is switched on, it starts reading and analyzing sector data, which are stored in its database. It is important to point out that the database must contain at least the basic sectors information (such as the geographic definitions) before the application starts, in fact the agent can not establish *a priori* what is the sector in which the device is at each time. When a sector is recognized, the system reads the corresponding sector information (i.e. the context definition). If no information about context are present, the system asks for them by using the web-services mechanisms (obviously the EGPRS/GSM device support has to be available) and connecting to the Service Center.

During the execution, whenever the system identifies a VG, it performs some actions related to the VG. In our scenario, the actions are the following:

- Display a graphic message on the PDA screen for a brief time.
- Activate text to speech engine to play a voice message.
- Send some information to the SC (such as the virtual gantry just crossed, the GPS timestamp and the mobile device unique identifier) for monitoring the mobile device travel.

---

[6] http://h10010.www1.hp.com/wwpc/it/it/ho/WF25a/22655-22657-22657-22657-298283-12343158.html
[7] http://msdn2.microsoft.com/en-us/sql/aa336364.aspx
[8] **J**ava **2** Platform, **E**nterprise **E**dition: http://java.sun.com/j2ee/1.4
[9] **K**eyhole **M**arkup **L**anguage: is a file format used to display geographic data in an Earth browser, such as Google Earth, Google Maps, and Google Maps for mobile. KML uses a tag-based structure with nested elements and attributes and is based on the XML standard.

By the combination of these three simple actions, the system realizes a complete satellite-based informative system.

Tests have been done selecting a particular sector and drawing some VGs over pre-defined local roads.

Figure 5a and **Figure 5**b show the Google Earth snapshots of a particular case of study: a sector is a zone of the Livorno province (Tuscany-Italy), and the selected road is a segment of the Livorno-Grosseto highway. After starting the test, we populated the mobile device database with the KML sector data. When the software starts, it recognizes the starting sector and asks the SC for context data. When context data has been received, the software analyzes and stores them in the mobile device database. After that, the mobile device is ready to run. Notice that the time spent for downloading and storing data operations is proportional to the number of geo-objects defined in the sector recognized by the agent and the cellular network throughput. In our experiments, by using the EGPRS[10] technology which permits a maximum of 25Kbits/sec transfer data, the time spent for downloading and storing the ninety virtual gantries defined in the context was about one minute. We executed several tests using ten mobile devices for a period of 15 days and we collected about one hundred car trips. We checked all travel data sent by the mobile devices to the SC and they resulted corrected. All defined virtual gantries had always been recognized by the agent, and the software worked properly also in hard working situations (for example, poor GSM signal or bad GPS quality). No abnormal conditions and faults have been detected. So the tests gave us a 100% of positive feedbacks.

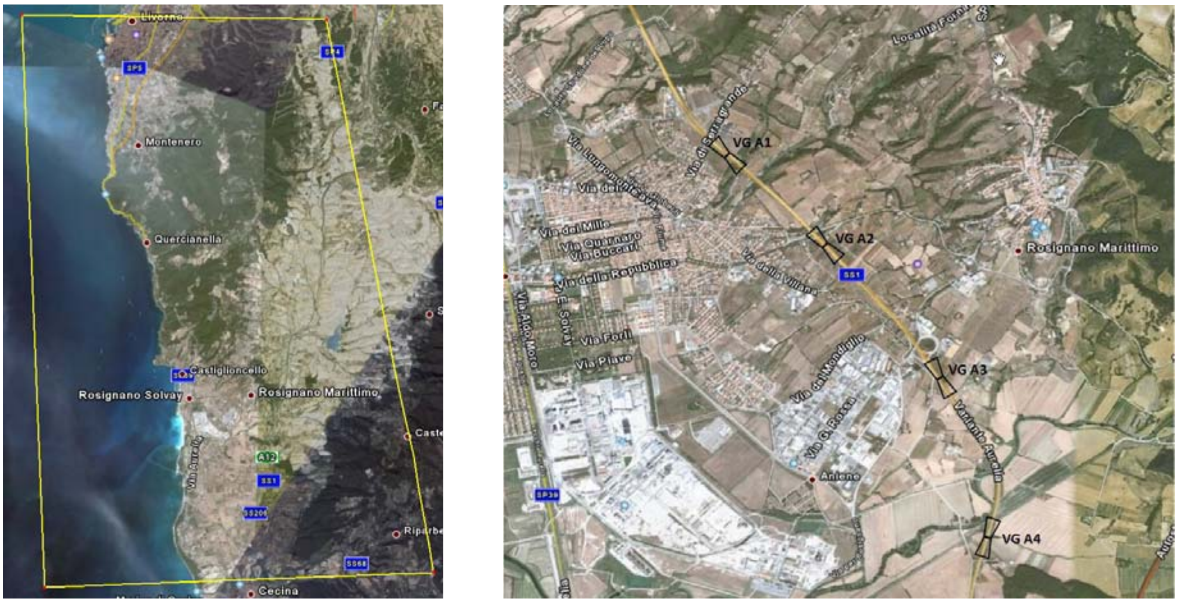

Figure 5. Screenshot from Google Earth,

(a) A geographical sector. (b) VGs drawn on the highway.

4. CONCLUSION

In this paper, we showed a system installed on mobile devices, for executing some actions dependently on the geographical location. The system is based on an **extensible** data model and permits a high level of interaction with the final users. The system can communicate to the SC some interesting information about the user path and some other pre-defined data. Furthermore it executes actions whenever a geographical event is recognized. These kind of actions are referred to the final user and can be of type: “visualize a graphical information”, “play an audio message”, “show the geographical position on the map”, … It is

[10] **Enhanced GPRS:** is a digital mobile phone technology that allows it to increase data transmission rate and improve data transmission reliability.

important to point out that the set of actions is extensible for modeling the user needs or for any kind of scenario in which the geographical position is the key factor to react upon.

A possible future work is to examine some methods for consolidating the GPS accuracy and GPS signal faults. In fact, the lack of GNSS[11] coverage in some environments is a real problem that cannot be solved with a GPS. At the moment, our prototype is tightly dependent to the quality and precision of the GPS hardware device and the GPS signal has to be present always (otherwise, the localization algorithm can establish where the mobile device is located). A GPS **dead reckoning** system will be desirable to resolve some GPS faults and to allow a continuous navigation. So, the mobile device can be completed by some hardware sensors, such as *odometers* and *gyroscopes*. In [8, 9, 10], you can see good approaches to integrate a GPS system with this kind of sensors based on an Extended Kalman Filter [11].

REFERENCES

Dickinson, P. et al, 2003. *ADO.NET Developers Guide*. Hoepli S.p.A, Milan - Italy.

Franson Technology, 2006. *Reference Manual Franson GPSTools .NET Compact Framework*. Franson Technology AB, Stockholm - Sweden.

Global Positioning System. *http://www.gps.gov*.

ISO, 2004. *Road transport and traffic telematics – Electronic Fee Collection – Application interface definition for electronic fee collection (EFC) based on Global Navigation Satellite Systems and Cellular Network (GNSS/CN)*. pp. 283-284; 335-336.

Loquendo, 2007. *Loquendo TTS Multilanguage Text-to-speech Synthesizer – SDK Programmer's Guide 6.6.* Loquendo S.p.A, Turin - Italy.

Fergus, D. and Roof, L., 2003. *The Definitive Guide to the .NET Compact Framework*. Apress, USA.

Google, 2007. *KML Online User Guide*. Google, USA.

Hu, Z., Wang, C. and Uchimura, K., 2007. Map learning and real-time vehicle localization for visual navigation. *Proceeding of International Conference on Mechatronics*. Kumamoto, Japan.

zur Bonsen, G. et al., 2005. Continuous navigation - Combining GPS with Sensor-Based Dead Reckoning. *GPS Word.*

B.Úbeda, R.Toledo, J.G.Jordán , S.Montes, 2004. *A theoretical and practical analysis of GNSS based road pricing systems, considering the EGNOS/SISNET contributions*. Universitad de Murcia, LISITT, Spain.

Simon, D, 2006. *Optimal state estimation: Kalman, H-infinity, and nonlinear approaches.* John Wiley & Sons, USA.

R.Maehara, 1984. *The Jordan curve theorem via the Brouwer fixed point theorem*, American Mathematical Monthly 91, no. 10, pp. 641–643, USA.

[11]GNSS: global navigation systems based on satellite positioning.