GPSIM: A Personal Computer-Based GPS Simulator System

D. Ibrahim

(*Traffic Control Systems Unit, London*)

Global Positioning Systems (GPS) are now in use in many applications, ranging from GIS to route guidance, automatic vehicle location (AVL), air, land, and marine navigation, and many other transportation and geographical based applications. In many applications, the GPS receiver is connected to some form of intelligent electronic system which receives the positional data from the GPS unit and then performs the required operation. When developing and testing GPS-based systems, one of the problems is that it is usually necessary to create GPS-compatible geographical data to simulate a GPS operation in real time. This paper provides the details of a Personal Computer (PC)-based GPS simulator system called GPSIM. The system receives user way-points and routes from Windows-based screen forms and then simulates a GPS operation in real time by generating most of the commonly used GPS sentences. The user-specified waypoints are divided into a number of small segments, each segment specifying a small distance in the direction of the original waypoint. The GPS sentence corresponding to the geographical coordinates of each segment is then sent out of the PC serial port. The system described is an invaluable testing tool for GPS-based system developers and also for people training to learn to use GPS-based products.

1. introduction. Currently, there are two satellite-based positioning systems in use. The American system, known as the *Navstar GPS* (standing for *Nav*igation *S*atellite *T*iming *A*nd *R*anging Global Positioning System), and the Russian system[1,2] named GLONASS (standing for *Glo*bal *Na*vigation *S*atellite *S*ystem). In this paper, only the Navstar GPS is considered, and the term GPS refers to the Navstar GPS system only.

GPS is a US military-based electronic navigation system[3,4] which became operational in the 1990s. The system is based on 24 satellites in six orbital planes using the NAVSTAR constellation and provides real-time position, velocity, and time information to users, free of charge, anywhere in the word, 24 hours a day. The GPS satellites orbit the Earth twice each day, at an altitude of 20000 km above the Earth and transmit precise position and timing data. A GPS receiver is a small electronic unit[5] which receives signals from the GPS satellites and calculates the position and elevation of the user. GPS receivers are available either as small hand-held units, or as electronic cards for OEM applications. The hand-held devices[6,7] are no bigger than the size of a hand-held calculator, and they can feature large LCD graphical displays. Card-type GPS units can be fitted into the back-plane of a standard PC bus, or they can be incorporated into dedicated OEM designs.

GPS satellites emit two types of signals:[8] C/A (Coarse Acquisition), and PPS (Precision Positioning System). Ordinary civilian GPS receivers can acquire and decode the C/A signals to give a position accuracy of typical 15 metres. However, the US Department of Defense has introduced a deliberate random error into the C/A signal, known as Selective Availability (SA). With this error, the horizontal position accuracy of about 100 metres can be obtained. The C/A vertical position (altitude) accuracy is 140 metres, and the time accuracy is within 340 nanoseconds. PPS signals are encrypted and can only be decoded by special military GPS receivers,[9] available only to licensed users.

GPS receivers provide RS232C type serial output signals, conforming to the National Marine Electronics Association[10] (NMEA) standards. A GPS with NMEA 0183 standard will output printable serial ASCII code, called the 'GPS sentences' to specify the position, altitude, time etc. of the GPS unit.

One of the problems with GPS-based system development is that, testing the system in a stationary place, such as a laboratory and also in real time requires test GPS data to be fed to the RS232C serial port of the system under test. Also, when training to learn the use of a GPS-based navigation system, it is necessary to provide simulated GPS data to the trainee's system. Some such systems operate by reading GPS data from a pre-prepared computer file. These systems have the disadvantage that they do not simulate a true GPS since the data are not received from the serial port. Also, it is not easy to modify the data pattern in such files. Some GPS receiver units[11] can be configured to operate in simulation mode where a journey along a route can be simulated with GPS sentences output from the unit. Although such receiver units can be used in some simulation applications, they have the disadvantages that the duration of the simulation cannot be controlled easily.

This paper describes the details of a GPS simulator program, called GPSIM, which is a prototype system, running on a PC. The program receives user waypoints from the user terminal and creates routes based upon these waypoints. The routes are then divided into as many segments as specified by the user. GPSIM then outputs from a serial port of the PC the most commonly used GPS sentences describing each segment. Thus, the simulator described in this paper simulates a true GPS environment in real time.

2. gps sentences. NMEA is a non-profit making association composed of manufacturers, distributors, dealers, educational institutions and other organizations interested in marine electronics. GPS communication standards are set and regulated by NMEA, and the standard NMEA-0183 defines the GPS input-output data protocols. Under this standard, all characters used are ASCII text, sent at 4800 baud (bits/sec). The data are transmitted in the form of sentences where each sentence starts with a '$' character and ends in a carriage-return line-feed pair. The '$' character is followed by a two character 'talker ID' (which is 'GP' for GPS receivers), a 'sentence ID', and a number of data items, separated by commas. An optional checksum field at the end of a sentence consists of a '*' identifier character and two hexadecimal characters, representing the exclusive OR of all the characters transmitted in a sentence, including the '$' and the '*'. Thus, the sentence format can be represented as:

$$\$GPggg, data\ 1, data\ 2, data\ 3, \ldots, data\ n, {*}cs\langle cr\rangle\langle lf\rangle$$

where, *ggg* is the sentence ID, *data 1* to *data n* are the data points, *cs* is the two digit checksum character, *cr* and *lf* are the carriage-return and line-feed characters respectively. If data for a field are not available, the field is omitted but the commas are included with no spaces between them. The data fields may be variable width, and a received sentence should be located and decoded by counting the commas and not by character positions.

https://doi.org/10.1017/S0373463398008170 Published online by Cambridge University Press

Some GPS sentences are so important that they are used by many manufacturers. The standard also allows individual manufacturers to define their proprietary sentences. These sentences start with a letter 'P', followed by a 3-letter manufacturer ID, and one or more characters which define the sentences. Table 1 lists some of the

Table 1. Some commercially available hand-held GPS receivers and the sentences they use (only the NMEA V2.0 standard sentences are given)

Manufacturer	Model	GPS sentences used
Garmin	GPS-38	GLL, RMB, RMC, WPL, BOD, GSA, GSV
Garmin	GPS-45	GLL, RTE, RMB, RMC, GGA, GSA, GSV, WPL
Garmin	12XL	GGA, GSA, GSV, RMB, RMC, WPL
Magellan	Trailblazer	APB, BWC, GGA, GLL, RMB, RMC, VTG
Trimble	Scoutmaster	APA, APB, BWC, GGA, GLL, GSA, GSV, RMB, RMC, VTG, WCV, XTE, ZTG
Lowrance	Global Map	APB, GGA, GLL, GSA, GSV, RMB, RMC
Eagle	Explorer	APB, GGA, GLL, GSA, GSV, RMB, RMC

commercially available GPS receivers and the sentences transmitted by these receivers.

It is not necessary for a GPS simulator to generate all of the sentences which are normally generated by a GPS. The sentences which are highly popular are the ones which give the geographical coordinates, time and date. The GPSIM simulator has been designed to generate the following three most commonly used sentences. All of these sentences define the geographical coordinates and the date and time.

*$GPGLL, lat, latdir, lon, londir, tim, v*cs*

*$GPRMC, tim, w, lat, latdir, lon, londir, sog, cmg, date, var*cs*

$GPGGA, tim, lat, latdir, lon, londir, fix, nsat, hdop, alt, hog, tdgps, idgps

The positional data (*lat*, *latdir*, *lon*, *londir*) and the date and time data (*tim*, *date*) are generated dynamically and continually. The remaining data items are assigned fixed values since they are not so important during the simulation of a GPS.

3. gpsim simulator system. There are several important requirements from a simulator of this type. Perhaps the most important requirement is that the simulator should be user-friendly and easy to use. Graphical user interface (GUI) type, Windows-based entry forms are very commonly used in most software applications today. A GUI enables users to enter and modify data easily and to make selections using a mouse. In addition to basic windows, such applications usually include menus, toolbars, command boxes, text boxes and so on. Another requirement is that the simulator should provide data in real time so that the operation of a GPS can be simulated. A GPS simulator should output most of the commonly used GPS sentences.

GPSIM consists of a standard laptop or desktop PC, equipped with one or more RS232C type serial ports, and a software, running in the foreground on the PC. The program is Windows-based and has been developed using the *Visual Basic* programming language[12] on a PC running under the *Windows 95* (or *Windows NT*) operating system. *Visual Basic* is an object-orientated programming language, used mainly to develop GUI-based applications programs which run under the Windows operating systems. The simulator program is menu-based and, when activated, the user is given the following choices (see screenshot in Figure 1):

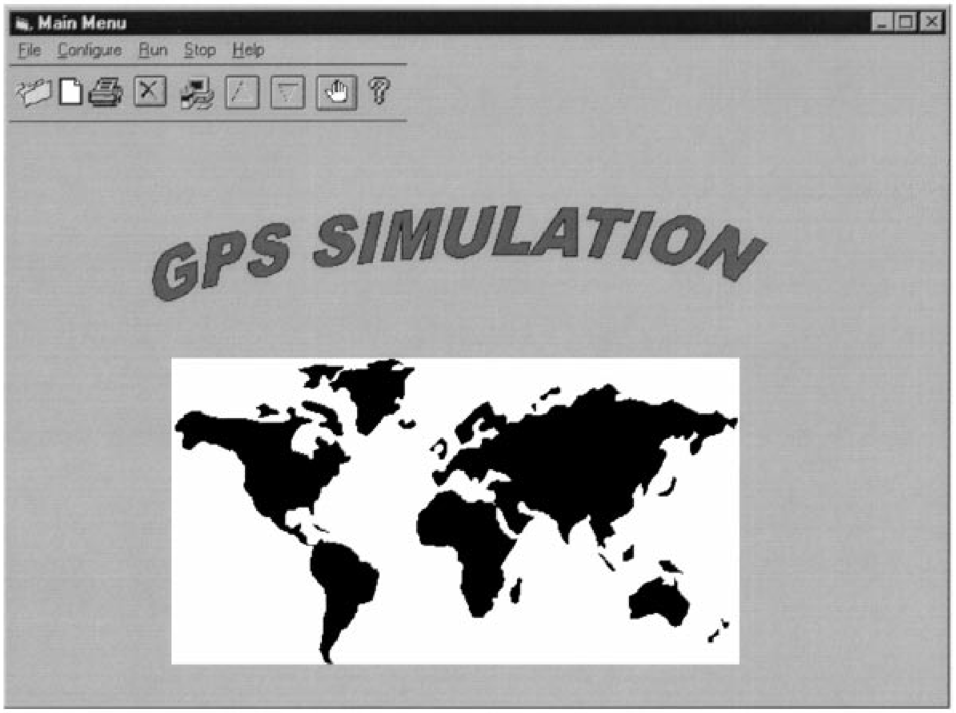

Figure 1. Screenshot of GPSIM main menu showing the menu items and toolbars.

(i) *File*
(ii) *Configure*
(iii) *Run*
(iv) *Stop*
(v) *Help*.

3.1. *File*. This is the route file processing option. Standard Microsoft command dialog boxes are used for file selection. When selected, the user is given the choice of creating a new route file (*New*), opening an existing route file for editing (*Open*), deleting a route file (*Delete*), printing a route file (*Print*), and terminating the program (*Exit*).

When a new file is to be created, or an existing file is to be opened for editing, the user is given a form as shown in Figure 2. With the aid of this form, the user enters the names of waypoints, the latitude and longitude of each waypoint together with their directions, and the number of segments that the distance between two waypoints are to be divided into during the simulation. Thus, this form completely defines a route in terms of its waypoints where each leg is to be divided into a number of segments during the simulation. The same form is used for both entering new waypoint data and also for modifying or deleting an existing waypoint. The *Delete*

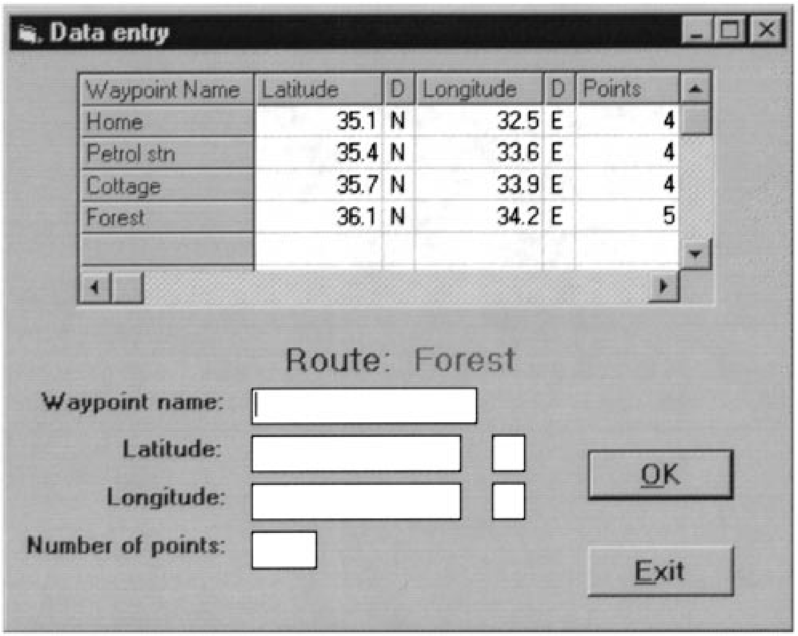

Figure 2. Screenshot showing GPSIM data entry form.

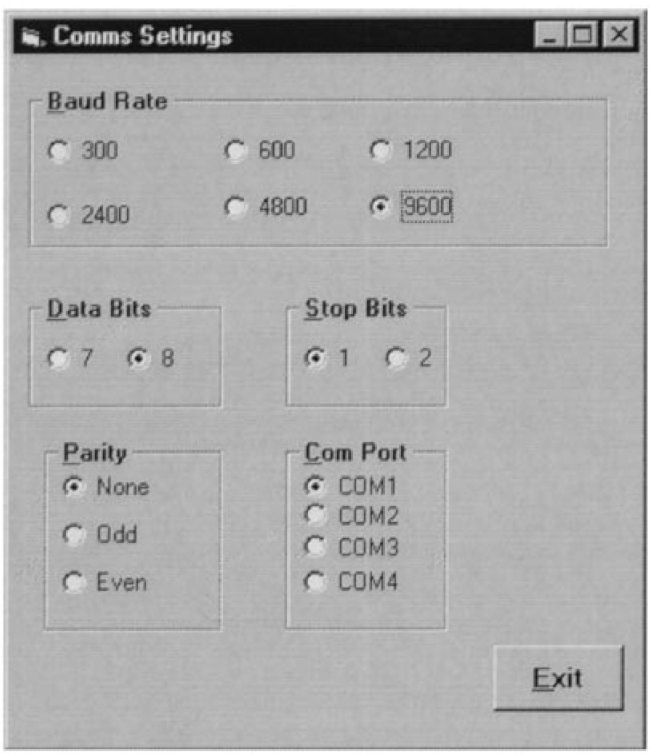

Figure 3. GPSIM communications setup menu.

https://doi.org/10.1017/S0373463398008170 Published online by Cambridge University Press

and the *Print* options enable the user to delete and print the contents of an existing route file respectively.

3.2. *Configure*. The *Configure* option enables the user to specify the RS232C serial GPS communication parameters. As shown in Figure 3, the baud rate, number of data bits, number of stop bits, parity, and the serial communication port number are entered by the help of this form. The data is saved in a file and then restored automatically when the program is activated.

3.3. *Run*. With this option the user starts the simulator. The simulation can be carried out in the normal forward direction from the first waypoint to the last. Alternatively, a reverse simulation can be performed where the simulation starts from the last waypoint and stops when the first waypoint is reached. The latter option is useful when simulating a return journey along a route. The time, in seconds between each equal waypoint segment is entered by the user so that real-time operation can be simulated as closely as possible.

The simulation of the journey requires the calculation of the length and the bearing of each waypoint segment. The length of each segment can be calculated once the total distance between two end points of a given leg is calculated. For short legs, it can be assumed that the bearing of each segment is the same as the bearing of the next waypoint. Given the length, starting coordinate and the bearing of a segment, the coordinates of the next segment can easily be calculated. These coordinates are then converted into GPS sentences and output from the serial port of the PC. The length and bearing of each segment is calculated as follows:

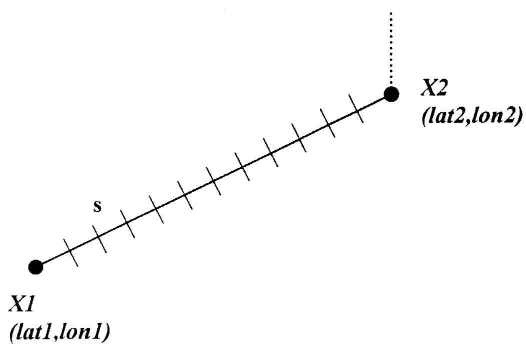

Figure 4. A leg X1 to X2 is divided into *n* segments, each having length *s*.

A given leg *X1* to *X2* is divided into *n* equal segments, each having a length *s*, as shown in Figure 4. Assume that (*lat1*, *long1*) and (*lat2*, *long2*) are the latitudes and longitudes of points *x1* and *x2* can be calculated using the standard great circle distance[13] formulae:

$$d = r \times a\cos[\sin(lat1) \times \sin(lat2) + \cos(lat1) \times \cos(lat2) \times (long1 - long2)] \quad (1)$$

where, *r* is the radius of Earth (6365 km)

Since the trigonometric function *acos* is not provided by Visual Basic, equation (1) is calculated using the *atn* function:

$$g = \sin(lat1) \times \sin(lat2) + \cos(lat1) \times \cos(lat2) \times \cos(long1 - long2) \quad (2)$$

and,

$$l = atn \quad \frac{\overline{1-g^2}}{g} \tag{3}$$

and,

$$d = r \times l \tag{4}$$

the distance calculated in equation (4) is divided into n equal segments where n is the number of samples required by the user. The length s of each segment is thus,

$$s = \frac{d}{n}. \tag{5}$$

The bearing θ of a given waypoint (and thus each segment within a leg) is calculated using the formula[13]:

$$b = \frac{\sin(lat2) - \cos(l) \times \sin(lat1)}{\sin(l) \times \cos(lat1)} \tag{6}$$

and,

$$\theta = \text{atn} \quad \frac{\overline{1-b^2}}{b}. \tag{7}$$

Once the length, starting coordinate, and the bearing of a segment is known, the coordinates of other segments are calculated as follows:

Assuming lat_i and $long_i$ are the latitudes and longitudes respectively of segment s_i, the coordinates of segment s_{i+l} are,

$$lat_{i+l} = lat_i + d\cos(\theta) \tag{8}$$

and,

$$long_{i+l} = long_i + d\frac{2\sin(\theta)}{\cos(lat_i + lat_{i-l})}. \tag{9}$$

The geographical coordinates calculated in (8) and (9) are formatted and used in the GPS sentences output by the simulator at the time intervals specified by the user.

3.4. *Stop*. The *Stop* menu option simply stops the simulation. This option is useful if the user wants to terminate the simulation because of errors or for some other reason. Once stopped, the simulator cannot be resumed, but has to be re-started from the beginning if required.

3.5. *Help*. *Help* option is an integral part of all modern GUI-based programs. Help is usually provided *online*, where the user can get instant assistance on various aspects of the programs they are using. A properly designed help option makes the software more friendly as the users can quickly become familiar with various options and features of the program. In complex applications, several other forms of online help are provided, including nested help menus, pop-up help menus, context-sensitive help, sample program codes, sample pictures etc, where the user is given the option of displaying or printing a hard copy of help topics. GPSIM provides help as a simple text file, describing the simulator commands in detail, where the user can scroll up or down the help file to access the help topics.

4. example output from gpsim. Perhaps the best way to show the various features of GPSIM is to look at a simple example. For this example, a route has been created consisting of four waypoints, chosen randomly between the lat 35·1° N

```
$GPGLL,3505.999,N,03230.000,E,045658,A*28
$GPRMC,045658,A,3505.999,N,03230.000,E,010.0,071.0,200298,003.9,W*6A
$GPGGA,045658,3505.999,N,03230.000,E,1,08,0.9,100.0,M,46.9,M,,*4B
$GPGLL,3509.656,N,03243.145,E,045703,A*23
$GPRMC,045703,A,3509.656,N,03243.145,E,010.0,071.0,200298,003.9,W*61
$GPGGA,045703,3509.656,N,03243.145,E,1,08,0.9,100.0,M,46.9,M,,*40
$GPGLL,3513.313,N,03256.301,E,045708,A*21
$GPRMC,045708,A,3513.313,N,03256.301,E,010.0,071.0,200298,003.9,W*63
$GPGGA,045708,3513.313,N,03256.301,E,1,08,0.9,100.0,M,46.9,M,,*42
$GPGLL,3516.969,N,03309.466,E,045713,A*24
$GPRMC,045713,A,3516.969,N,03309.466,E,010.0,071.0,200298,003.9,W*66
$GPGGA,045713,3516.969,N,03309.466,E,1,08,0.9,100.0,M,46.9,M,,*47
$GPGLL,3520.626,N,03322.641,E,045718,A*20
$GPRMC,045718,A,3520.626,N,03322.641,E,010.0,071.0,200298,003.9,W*62
$GPGGA,045718,3520.626,N,03322.641,E,1,08,0.9,100.0,M,46.9,M,,*43
$GPGLL,3524.000,N,03335.999,E,045723,A*22
```

Figure 5. Sample output from the simulator.

long 32·5° E and lat 36·1° N, long 34·2° E. The distance between each waypoint is selected as 4 segments; thus, the route will be simulated with 16 points, and each point shall be described by three GPS sentences, making a total of 48 sentences. Figure 2 shows the input form used to enter the names, coordinates, and the number of samples required between each leg. The time interval between each output sample was chosen as 5 seconds, thus the total simulation time was 80 seconds (for 16 points). The simulation was run in forward mode, and part of the data output from the simulator is shown in Figure 5 as ASCII text of GPS sentences. This data is output from the serial PC port and is also saved in a sequential disk file.

5. conclusions. One of the nice features of GPSIM is that the simulator runs on a standard laptop or desktop PC. The simulator data is output from a serial port of the PC. The simulator can be used in three different configurations, depending upon the availability of serial ports on the PC used.

(i) If the PC is equipped with more than one serial port, then the simulator and the application which requires the simulator data can run on the same PC. As an example, the simulator can be configured to output to serial port 1 and port 2 can be configured for the application program. Port 1 can then be connected to port 2 so that the GPS simulator data can be fed to the application program.

(ii) If, on the other hand, the PC is equipped with only one serial port, then a PC (e.g. a laptop) can be used to output the simulator data, and this port can be connected to the serial port of another PC where the application program is running.

(iii) The other option is to design an electronic buffer circuit which can be connected to the serial port of the PC to receive and store the simulator data in a non-volatile memory. This buffer circuit can then be used wherever required to provide GPS simulated data.

The user also has the option of saving the output of the simulator in a sequential text file. Any application program can read the simulator data from this file. Since the file contains only the GPS sentences, the application program will have to provide the required delay between the sentences so that real-time simulation can be achieved.

GPSIM can be enhanced by the addition of more features. One such enhancement could be the inclusion of more standard and proprietary GPS sentence types and also the simulation of the other data types (e.g. satellite availability, sog, cmg etc.), not currently used in GPSIM. Map-based data entry could also be incorporated into GPSIM so that waypoints can be selected directly on the map without the need to use Windows-based data entry forms. This has the advantage that it will eliminate the need to find the waypoint coordinates manually.

nomenclature

alt	altitude
hdop	horizontal dilution of position
lat	latitude
latdir	latitude direction
long	longitude
londir	longitude direction
nsat	number of satellites tracked
Idgps	DGPS station id number
w	GPS navigation receiver warning
hog	height of geoid
sog	speed over ground
cmg	course made good
var	magnetic variation
fix	GPS fix type
tim	time
v	GPS checksum
tdgps	time since last DGPS

acknowledgements

The author wishes to thank all the GPS manufacturers who provided information on their products. The ideas presented in this paper are those of the author only and do not represent the ideas or policies of any organisation or body.

references

[1] Daly, P. (1988). Aspects of the Soviet Union's GLONASS satellite navigation system. This *Journal*, **41**, 186–198.

[2] Dale, S. A., Kitching, I. D. and Daly, P. (1988). Position-fixing using the USSR's Glonass C/A Code. *IEEE Position Location and Navigation Symposium*, pp. 13–20.

[3] Ackroyd, N. and Lorimer, R. (1990). *Global Navigation: A GPS User's Guide*. Lloyd's of London Press Ltd., London.

[4] Logsdon, T. and Helms, C. W. (1984). *The Navstar GPS*. A status report. Paper read at the 5th Annual Armed Forces Communications and Electronics Assoc. Symposium and Exposition, 24 October, Brussels, Belgium.

[5] Pratt, T. (1989). The Electronics and Hardware of GPS receivers. *Seminar on the Global Positioning Systems*. University of Nottingham 12–14 April.

[6] Nordwall, B. D. (1994). Small GPS receivers open new possibilities. *Aviation Week and Space Technology*, pp. 57–58.

[7] GPS World Survey (1995). *GPS World*. January, pp. 50–67.

[8] Logsdon, T. (1995). *Understanding the Navstar*. Van Nostrand Reinhold Publishing, New York.

[9] Aggarwal, A. K. (1988). EURONAV – A state-of-the-art military GPS receiver. *IEEE Position Location and Navigation Symposium*, pp. 153–164.

[10] National Marine Electronics Association (NMEA), PO Box 3435, New Bern, USA.
[11] *GPS-45 Personal Navigator Owner's Manual & Reference*. Garmin International, 9875 Widmer Road, Lenexa, KS 66215, USA.
[12] Roof, L. (1996). *Visual Basic 4 Professional*. Wrox Press Ltd., Chicago, USA.
[13] Kennewell, J. (1997). Great Circle distances and bearings. *IPS Radio* & *Space Services*, Sydney, Australia.

key words

1. GPS. 2. GPS Simulator. 3. PC-Based Simulator.