On-line GPS Track Simplification Algorithm for Mobile Platforms

R. Ivanov

Key Words: *GPS track simplification; GPS navigation; Kalman filters.*

Abstract. *The paper presents an algorithm for on-line simplification of the number of points, describing a GPS track. The proposed algorithm is adaptive to the accuracy of the GPS receiver, the current accuracy of determining the position and the speed of movement. An on-line simplification of the number of track points is offered on the base of analysis of the location of three last points. The degree of reduction of the points is from 3 to 15 times, depending on the track length and trajectory. The errors in the determination of the GPS position are reduced as the values for the latitude and longitude are filtered by an adaptive Kalman filter. The GPS position filtering results in up to 15% additional reduction of number of points in pedestrian mode.*

1. Introduction

The most of existing GPS navigators and mobile applications for GPS navigation enable the record of a track of the path passed. For that purpose, the route passed is composed as a sequence of points, describing the current client's location. A track may be recorded in one of several formats, such as: GPS eXchange (GPX), Keyhole Markup Language (KML), Comma-Separated Values (CSV), and etc.

The main problems in the creation of a track are:

- Reduction of the number of the points, that describe the track, without losing important information;
- The tracking algorithm realization will not depend on the speed of movement (hiking, running, cycling, boating, and etc.);
- Filtering the points that do not describe the track accurately, due to the insufficient temporary accuracy of the GPS receiver.

In the development of tracking algorithms for mobile terminals it is necessary to take into account the additional limitations as well, such as:

- Optimization of the operative memory used due to its limited capacity;
- Usage of algorithms that allow for the realization of the tracking in real time without significant battery discharging and processor power consumption.

2. Related Work

The GPS navigators for automobiles which support tracking mode impose limitations on the number of tracks and the number of points, that describe every track. For example, for Garmin GPS nьvi series 750, 760 the number of the tracks is up to 10, and for Garmin GPS 7200, Quest – up to 50. The number of the points is most often up to 10000. The situation with the GPS navigators for hiking (Garmin GPS eTrex, Legend) is similar.

In the mobile applications for GPS navigation the number of the tracks depends mainly on the available flash memory and the number of the points – on the free operative memory and the realization of the tracking algorithm. Most widely used are the next following approaches for selection of the moment to enter a new track point [14]:

- After a preset time interval – time-based tracking algorithms.
- After a preset distance passed – distance-based tracking algorithms.

Actually, the moment of entering a new track point depended also on:

- Travel speed.
- The current accuracy of the GPS receiver.
- The route trajectory (straight segment, curve).

Table 1. Tracking algorithms comparison

Application and platform	Tracking algorithm
PDA.bgmaps [11] Microsoft Pocket PC	Distance-based (up to 3600 points per track)
gpsVP [10] Windows Mobile	Distance-based
Odgps 1.2 [17] Pocket PC 2003, J2ME	Time-based turn-rate based
AFTrack 1.20 [13] Symbian (S60)	Manual mode time-based distance based turn-rate based
GPS Track [15] J2ME	Time-based distance based
GPS Mid [9] J2ME	Time-based distance based adaptive to speed
Nokia Sport Tracker [12] Symbian (S60)	Simple filter is added to filter out erroneous GPS locations
Mobile Trail Explorer [16] J2ME	Time-based distance based

Table 1 describes the parameters of tracking algorithms of analyzed GPS navigation applications for mobile phones.

In all applications the user must enter the time interval or the path passed after which a new track point is generated. In two of the applications the moment of a new point generation depends adaptively on the travel speed.

3. Algorithm Description

The proposed tracking algorithm belongs to the distance based algorithms, but the threshold value for the distance after the passing of which a new point is entered is obtained adaptively, taking into account: the current accuracy of user's position and the travel speed. An additional reduction of the number of points is realized by means of analysis of the position of the last 3 points generated.

Figure 1 shows the sequence to obtain the moment to enter a new track point.

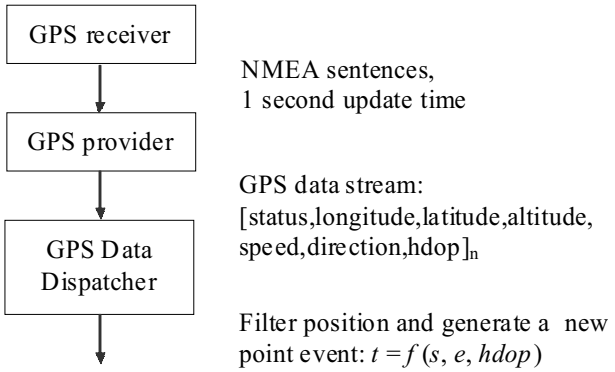


Figure 1. Sequence to obtain the moment to enter a new track point

The data from the GPS receiver are processed by the program module „GPS Provider", which parses the National Marine Electronics Association (NMEA) 0183 sentences to the necessary GPS data. The module „GPS Data Dispatcher" is intended to adaptively define the moment of generation of a new track point. The time interval (t), after which a new point is entered, depends on: traveled distance (s), the error in the user position (e), and the current horizontal accuracy of GPS receiver ($hdop$).

3.1. Adaptation to the Maximum Positioning Error

The maximum error in the user's position (e) depends on the accuracy of the GPS receiver (E), and on the current latitude and longitude accuracy, which depends on the number of visible satellites and the availability of reflected signals. To evaluate the current accuracy we will use the value of the parameter Horizontal Delusion of Precision ($HDOP$)

(1) $e_n = E.hdop_n$, [m].

The experiments show that in order to guarantee the accuracy of a GPS navigation system $HDOP$ must be better than 4.5 for movement by car, and better than 3.5 when walking. The movement mode (by car or on foot) is software-defined, analyzing the speed trend. There–fore the speed values (v_n) are filtered by a first order low-frequency Infinite Impulse Response (IIR) filter:

(2) $\hat{v}_n = \alpha \hat{v}_{n-1} + (1-\alpha) v_n$, [km/h]

where:

v_n is the value of the speed in discrete moment n;

$\hat{v}_n$ is the filtered value of the speed in the discrete moment n;

α is a parameter defining how strongly the output reaction of the filter depends on the current speed value.

By speed filtering a smooth time switch between the two modes is obtained. It is assumed that if $\hat{v}_n > v_{Th}$ the travel mode is by car, otherwise – on foot.

3.2. Adaptation to the Travel Speed

The moment to enter a new track point depends on travel speed – when the speed is higher the probability of changing the directions of movement is lower than in the other case.

It is proposed a new track point to be entered after the user has passed a certain distance (s), the value of which changes adaptively depending on the travel speed.

It is assumed that the interval of possible values of s is within the range [e, S_{max}], where e is the current accuracy of positioning:

(3) $s_n = S_{max}$[m] if $\hat{v}_n$ > 100 km/h, otherwise:

$$s_n = e_n + \frac{S_{\max} - e_n}{100} \hat{v}_n \text{, [m]}$$

3.3. Wild Points Filtering

The largest error in GPS data is obtained when the accuracy of the GPS receiver is low and when changing the number of visible satellites and speed. This type of errors (wild points) must be filtered before the algorithm for track simplification. Wild points filtering is realized by blocking the generation of GPS data when there are significant changes in the number of visible satellites or in the values of HDOP and speed. For this purpose, program timers are used.

3.4. Erroneous GPS Positions Filtering with Adaptive Kalman Filter

The Kalman filtering is an optimal estimation method that has been widely used in Global Positioning and Inertial Navigation Systems [5]. For GPS navigation the level of measurement noise in GPS data is environmental and user movement dependent. In this case Kalman filter is not optimal and may cause to an unreliable results, for example over-smoothing in curves. One solutions for that problem is adaptive Kalman filtering. Adaptive filtering is trying to determine the statistic parameters of the dynamic system, based on the value of the predicted residuals [1]. To identify the vehicle dynamics and noise covariance matrix a fuzzy logic and neural networks are used [2,4,7]. Such techniques can not be realized in real time when application is started on mobile terminals.

The main source of errors in the definition whether a point shall belong to the track or not is the error in user's GPS position. It is biggest in the moment of transition from invalid to valid data. In order to minimize it we suggest the sequence of latitude and longitude values to be filtered through a first order

Kalman filter. The degree of filtering is defined through the values for the noise variance in the GPS data ($r_{\min} \le r \le r_{\max}$). We propose the limit values of r to depend on the current speed value (in curves the value of the filtered speed slowly falls, thus preventing over-smoothing):

$$\Delta v_n = floor(\hat{v}_n/4),$$

(4) $r_n = r_{\min}$ if $\hat{v}_n > 60$ km/h,

$$r_n = r_{\max} - \frac{r_{\max} - r_{\min}}{4} \Delta v_n \text{ otherwise.}$$

The value of r is re-calculated only if:

(5) $\Delta v_n \neq \Delta v_{n-1}$.

The algorithmic realization of the filter is as follows:

Initialization:

(6) $K_0 = 0$, $\hat{x}_0 = z_0$, $P_0 = 0$, $P_0^- = 1.0$,

$\hat{x}_0^- = z_0$, $q = 1.0$, $r = r_{\min}$

Prediction:

(7) $\hat{x}_k^- = \hat{x}_{k-1}$, $P_k^- = P_{k-1} + q$

Update:

(8) $K_k = P_k^- / (P_k^- + r)$

$\hat{x}_k = \hat{x}_k^- + K_k (z_k - \hat{x}_k^-)$

$P_k = (1 - K_k) P_k^-$

where $\hat{x}_k$ is a posteriori state estimate at discrete time k, $\hat{x}_0^-$ is a priori state estimate, z_k is actual measurement (longitude, latitude), P_k is a posteriori estimate error covariance, P_k^- is a priori estimate error covariance, K_k is Kalman gain, r is measurement noise variance, q is system noise variance.

The filtered data (longitude, latitude) coincide with the a posteriori state estimate.

3.5. On-line Simplification of Number of Points, Describing a Track

When Java 2 Mobile Edition (J2ME) based GPS navigation systems for the mass consumers are developed, optimization of the operative memory used is necessary. On-line reduction of the number of points in track is offered on the base of analysis of the location of the three last points entered (see *figure 2*).

The track simplification algorithm that is used belongs to perpendicular-distance algorithms.

Point p_1 is the last one belonging to the track. Point p_3 is the last generated point. It should be defined whether point p_2 belongs to the track. For that purpose the value of the angle γ, $\gamma = |\alpha_1 - \alpha_2|$ is obtained. If $\gamma \ge \gamma_{Th}$, then it is assumed that point p_2 is a part of the track, otherwise point p_2 is ignored. The value of the threshold γ_{Th} is obtained adaptively, depending on the value of the filtered speed:

(9) $\gamma_{Th} = \gamma_{\min}$ if $\hat{v}_n > 50$ km/h

$$\gamma_{Th} = \gamma_{\max} - \frac{\gamma_{\max} - \gamma_{\min}}{50} \hat{v}_n \text{ otherwise.}$$

In order not to miss important track points in long and smooth curves an additional check is made for the distance (d) between point p_2 and the segment, formed by points p_1 and p_3 (nearest path to track). If $d > d_{Th}$, point p_2 belongs to the track notwithstanding that the condition $\gamma \ge \gamma_{Th}$ is not fulfilled. The value of the threshold d_{Th} is obtained adaptively, depending on the current value of *HDOP*.

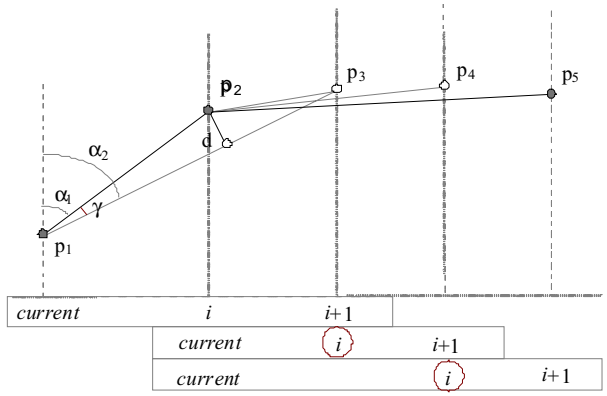


Figure 2. Reduction of the number of points, describing a track

3.6. Algorithm Pseudo-code

The module GPS Data Dispatcher is intended to inform via messages the other program modules for new events from the GPS receiver. *Figure 3* shows the main program loop through which the moment for generation a new point is determined.

1. Initialization:
   E=10, lastLon=0, lastLat=0, distTh=0, $\hat{v}_{n-1}$ =0
2. Get new GPS data:
   {lon, lat, alt, speed, direction, hdop, status}$_n$
3. AddNewPoint(GPS data)
4. Go to step 2

Figure 3. GPS Data Dispatcher main loop

The method AddNewPoint realizes: speed filtering, position filtering, travel mode definition (by car or on foot), and generation of a new point, if there are nece–ssary conditions – *HDOP*, the speed and path passed within the needed ranges (see *figure 4*).

The program module which realizes the tracking algorithm must listen its mailbox for „newpoint“ messa–ge. After receipt the message the GPS data are read by the GetNewPoint method and the reduction of the points is realized with the OptimizeTrack method (*figure 5*).

The optimization procedure requires two buffers of the type double (xpath, ypath) for buffering the coordinates of the last three points. The pseudo-code of the OptimizeTrack method is shown in *figure 6*.

```
1.  Algorithm AddNewPoint (GPS data)
2.    v_n =speed_n,  v̂_n = FilterSpeed( v̂_{n-1} , v_n )
3.  if ( v̂_n > v_Th )
4.     HDOP_max=4.5
5.  else
6.     HDOP_max=3.5
7.  endif
8.  if (hdop_n < HDOP_max)
9.     e_n = E.hdop_n
10.    if ( v_n >100)
11.       distTh = S_max
12.    else
13.       distTh = e_n + [(S_max − e_n )/100]v̂_n
14.    endif
15.    [filtLon_n, filtLat_n] = filterPosition(lon_n,lat_n)
16.    dist = GPSDist(filtLon,filtLat,lastLon,lastLat)
17.    if (dist ≥ distTh)
18.       lastLon = lon_n
19.       lastLat = lat_n
20.       POBox.add("newpoint")
21.    endif
22. endif
```

Figure 4. AddPointToTrack algorithm

```
1.  i=0,numberOfPoints=0,xpath[3],ypath[3]
2.  while
3.       GetNewPoint(filtLn_n, filtLat_n, hdop_n)
4.       OptimizeTrack(filtLn_n, filtLat_n)
5.  endwhile
```

Figure 5. Tracking algorithm main loop

Through the AddPoint method a new track point is inserted. The method is called only if the CheckPoints method, which realizes the algorithm for optimization of the number of points, returns *true*. The realization of the CheckPoints method is shown in *figure 7*.

4. Experimental Results

In order to evaluate the characteristics of the algorithm presented two types of experiments were made:

1. Off-line experiments with Matlab™: obtaining of values of the parameters used, adaptive Kalman filtering, track simplification, and analysis of the positioning error in comparisons between the optimal track, obtained by a GPS map and the test tracks, obtained on-line.

2. On-line experiments with mobile terminals (Nokia 6630, Nokia N95): record tracks in CSV format.

The programs developed for Matlab™ enable: parameter adjustment; obtaining optimal track in CSV format via GPS map; determination of the positioning error by comparison between the optimal track and the tracks, obtained when proposed algorithm is used.

In *table 2* are given values of the parameters used. They are obtained after off-line experiments.

```
1.  Algorithm OptimizeTrack(lon_n, lat_n)
2.  xpath[i] = lon_n , ypath[i] = lat_n
3.  i = i + 1
4.  if (i = 1)
5.     AddPoint(xpath[0], ypath[0])
6.     numberOfPoints++
7.     return
8.  endif
9.  if (i = 3)
10.    if (CheckPoints(xpath,ypath,hdop_n) = true)
11.       xpath[0] = xpath[1], ypath[0] = ypath[1]
12.       AddPoint(xpath[0], ypath[0])
13.       numberOfPoints ++
14.       return
15.    endif
16.    xpath[1] = xpath[2], ypath[1] = ypath[2]
17.    i = 2
18. endif
```

Figure 6. OptimizeTrack algorithm

```
1.  Algorithm CheckPoints(xpath, ypath, hdop_n)
2.  α_1 = Bearing(xpath[0],ypath[0],
              xpath[1],ypath[1])
3.  α_2 = Bearing(xpath[1],ypath[1],
              xpath[2],ypath[2])
4.  γ = abs(|α_1 − α_2|)
5.  if ( γ≥γ_Th )
6.     return true
7.  else
8.     dist = GetNearestDistToTrack(xpath,ypath)
9.     if (dist > d_Th )
10.       return true
11.    endif
12. endif
13. return false
```

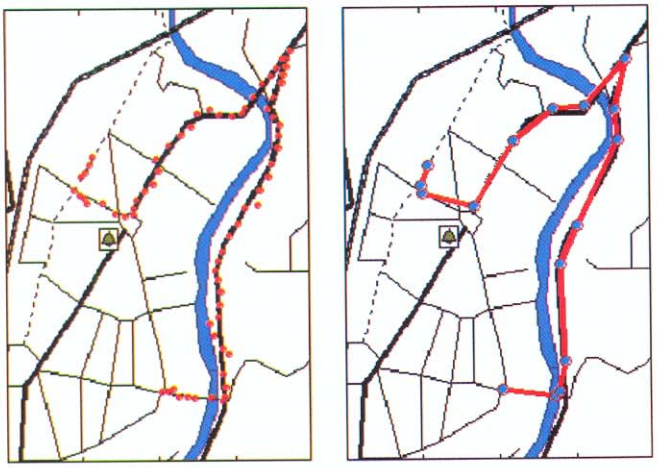
Figure 7. CheckPoints algorithm

Table 2. Parameter values

Parameter	Description
$\alpha = 0.65$	This value guarantees a delay of 8 sec. in case of route curves. The aim is to prevent over-smoothing due to high values of the parameter r.
$r_{\min} = 1.0$ $r_{\max} = 4.0$	The so selected interval for the parameter r guarantees the filtering of the GPS position without occurrence of invalid data.
$\gamma_{\min} = 10°$ $\gamma_{\max} = 25°$	The minimum and maximum value for the threshold for angle γ over which it is assumed that the point belongs to the route.
E= 10 m, d_{Th} = 4.25*hdop, $S_{\max}$ = 65 m, v_{Th} = 8km/h	

Figure 8a shows the track, which consists of 71 points, length 1775 m. After filtering and number of points reduction, a

track with 16 points (length 1738 m) is obtained. The actual length of the track is 1701 m.



a) b)

Figure 8. Example for track filtering and simplification

To calculate positioning error a track with two sections is analyzed: 1st section – car mode (the first 8 points) and 2nd section – pedestrian mode (the last 6 points). The total length of the optimal track is 1951 m and is described with 14 points. The track has been passed 20 times with *HDOP* values within the interval [0.9, 3.2]. Each track is recorded on the flash disk of the mobile terminal in CSV format. The trajectories of the optimal track and one of the test tracks are shown in *figure 9.*

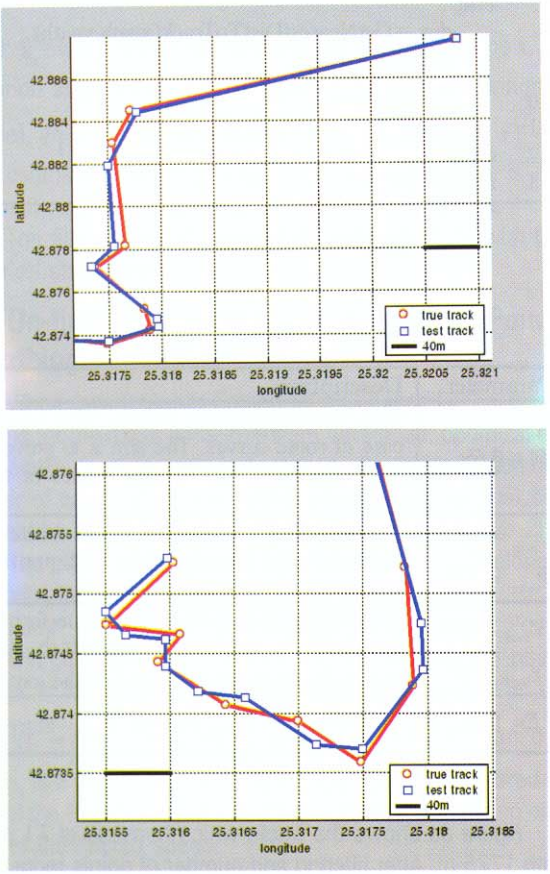


Figure 9. Original (true) and test track

The comparison between the optimal and test tracks is realized with a Matlab™ application. The results obtained are described in *table 3.*

Table 3. Experimental results – tracks comparison

Track	L, m	MPE, m	Nred/Nopt	Nraw/Nred
on car	1659	5.9	1.075	4.94
by foot	292	7.8	1.417	1.58
all	1951	6.85	1.221	3.26

The following abbreviations are used: L – track length; MPE – maximum positioning error; Nopt – number of points in optimal track; Nred – averaged number of points in test tracks, when the algorithm for on-line reduction of point number is active; Nraw – averaged number of points in test tracks, when the algorithm for on-line reduction of point number is not active.

The tracks, necessary for the testing of the algorithm, are obtained in CSV format via the J2ME application for GPS navigation. *Figure 10* shows two snapshots of the application: a list of the saved tracks, and a loaded track.

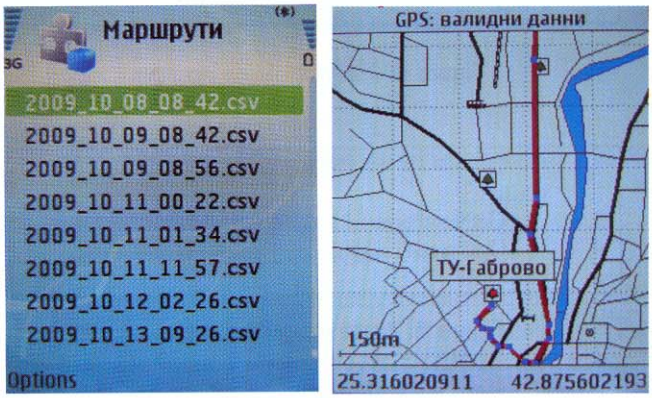


Figure 10. List of saved tracks and test track

5. Conclusions and Outlook

One of the major problems in the development of tracking algorithms for mobile platforms is the simplifycation of the number of points, describing the track. In most cases the optimization can be realized after entering the last track point. In this case, more complicated line simplification algorithms can be used (modifications of the Ramer-Douglas-Peucker algorithm [8] or kind of Evolutionary Computation algorithms [6]), but the optimizaion of the memory and on-line tracking are not possible.

An algorithm for on-line simplification of the number of points, describing a track, is presented. The algorithm is adaptive to the current accuracy of the GPS receiver, the speed and trajectory of movement. The decision for a new point is taken on the base of analysis of the position of the last 3 points generated. The track length and maximum number of points depend on the available free flash memory and the battery capacity. For Nokia N95 mobile terminal the battery discharging time is about 6.5 hours of continuous operation of the tracking algorithm. The degree of reduction of the points is from 3 to 15 times, depending on the track length and trajectory. The filtering

of the GPS position by means of a Kalman adaptive filter results in up to 15% additional reduction of number of points in pedestrian mode.

The algorithm presented is a part of a Java mobile application for outdoor navigation for visually impaired people [3]. The application does not require adjustment of parameters and user interaction.

Our future work will mainly focus on the more accurate definition of GPS data reliability, obtained by knowledge-based and fuzzy rules.

References

1. Congwei, H., et al. Adaptive Kalman Filtering for Vehicle Navigation. – *Journal of GPS*, 2, 2003, No. 1, 42-47.

2. Dah-Jing Jwo, Cheng-Min Huang. An Adaptive Fuzzy Strong Tracking Kalman Filter for GPS/INS Navigation. 33rd Annual Conference IECON 2007, 2007, 2266-2271.

3. Ivanov, R. Mobile GPS Navigation Application, Adapted for Visually Impaired People. International Conference Automatics and Informatics '08, Sofia, 2008, VII-9 to VII-12.

4. Kaygisiz, B. H., A. M. Erkmen, I. Erkmen. GPS/INS Enhancement Using Neural Networks for Autonomous Ground Vehicle Applications. Conference on Intelligent Robots and Systems, 2003, 3763- 3768.

5. Mohamed, H., K. P.Schwarz. Adaptive Kalman Filtering for INS/GPS. – *Journal of Geodesy*, 73, 1999, 193-203.

6. Serrano, J. I., J. Alonso, M. D. Castillo, J. E. Naranjo. Evolutionary Optimization of Autonomous Vehicle Tracks. IEEE Congress on Evolutionary Computation, 2, 2005, 1332-1339.

7. Wang, J., J. J. Wang, D. Sinclair and L. Watts. A Neural Network and Kalman Filter Hybrid Approach for GPS/INS Integration. 12th IAIN Congress & 2006 Int. Symp. On GPS/GNSS, Jeju, Korea, 2006, 277-282.

8. http://en.wikipedia.org/wiki/Ramer-Douglas-Peucker_algorithm

9. http://gpsmid.sourceforge.net

10. http://gpsvp.com

11. http://pda.bgmaps.com/index.aspx

12. http://sportstracker.nokia.com

13. http://www.afischer-online.de/sos/AFTrack/

14. http://www.guod.net/2007/5/9/a-review-of-route-reduction-track-simplification-algorithms

15. http://www.qcontinuum.org/gpstrack

16. http://www.substanceofcode.com/software/mobile-trail-explorer

17. www.outdoor-gps.de

Manuscript received on 20.10.2009

Rosen Ivanov *received the M.Sc. degree in electronic engineering and microelectronic from Technical University of Gabrovo and Ph.D. degree in automatic systems for signal processing and control (distributed speech recognition) from Technical University of Sofia in 2001.*
Hi is currently an associate professor with the Department of Computer Systems and Technologies, Technical University of Gabrovo. His research interest include digital signal processing, mobile communications and network programming.

Contacts:
Technical University of Gabrovo
4 H. Dimitar Str
5300 Gabrovo, Bulgaria
e-mail: rs-soft@ieee.org