A&QT-R 2002 (THETA 13)
International Conference on Automation, Quality and Testing, Robotics
May 23-25, 2002, Cluj-Napoca, Romania

GPS REAL-TIME CAR NAVIGATION SYSTEM

Prof.dr.eng Gavril Toderean, Technical University Cluj-Napoca
Eng. Ioan Toma, Technical University Cluj-Napoca
Information Scientist Gabriela Chira, Pro 3 Soft Cluj-Napoca
Eng. RaduVescan, Technical University Cluj-Napoca
Eng. Cristian Jurca, Technical University Cluj-Napoca
Eng. Alin Inclezan, Technical University Cluj-Napoca
Eng. Lucia Locuratolo, Metriqs, Ivrea (TO) Italy
Eng. Antonio Serra, Metriqs, Ivrea (TO) Italy
Eng. Marco Contenti, Metriqs, Ivrea (TO) Italy

Email: toderean@cluj.astral.ro
Web: http://pages.astral.ro/pro3soft

ABSTRACT

TNAV is an Activ X component, developed in Visual C++ 6.0, using MFC (Microsoft Foundation Classes). The TNAV (Turn-by-turn NAVigator) ActiveX control displays turning point information to the driver, based on route, data and GPS position. Route data is supplied every time a new route is calculated or selected, while GPS position is supplied with a regular sampling interval, when available.

The presented software is the result of the collaboration with the Italian Metriqs company.

KEYWORDS: GPS real-time car navigation system

1. INTRODUCTION OF THE GPS REAL-TIME CAR NAVIGATION SYSTEM

The idea of turn-by-turn navigation is now used in many navigation tools. Such tools do simplify the navigation of cars and people. We developed a control that, if integrated in an application, guides the driver of the car, from a start point to the destination, displaying parts of the route and other useful information to the driver.

The control that we developed works hand in hand with another control. This other control is named “PalmGeo”. It was developed by MAROS company, and calculates the route from the source to the destination. Different calculation types may be selected: the shortest path, the fastest path, and the path that excludes highways. The control displays the map and the route from the source to the destination. It also displays, every second, the position of the car on the route. One important task of the control is to write the structure of the route in a .xml file. A list of segments and a list of

streets make up the route, from the source to the destination. A segment structure contains a list of points, the name of the street to which it belongs, its length, and its time. The segment also contains a list of branches, if there is a crossroad at the beginning of the segment. Every branch contains an angle value. A point structure contains the GPS coordinates: Latitude and Longitude [1], [2].

2. DESCRIPTION OF THE GPS REAL-TIME CAR NAVIGATION SYSTEM

The TNAV control receives the information about the route in a Variant structure.
This information is received only one time at the beginning of the journey. If the user wants to change the route, this structure is also changed.

The TNAV Control receives every second the GPS coordinates of the car.

The graphic interface of TNAV Control is divided into three parts.

- The first part (area) is a progress area that contains a progress bar.
- The second part (area) is a display area .
- The third part (area) is a data area .

Using the variant structure, that contains the data for the journey and the GPS coordinates received every second, the TNAV control draws the 3 areas. For every new GPS coordinates received, the TNAV control takes some decisions. The process of taking those decisions is important, because some variables are sets, variables that determine the way the control displays the information.
These variables are:

1. **SegmentsNo -** the number of route segments that have to be displayed
2. **DisplayStartSegment** - the first segment of the route segments that are displayed.
3. **FirstSig -** the first significant segment of the segments that are displayed.

The display area and the data area do display different information, depending on the variable **SegmentsNo** (the number of segments that have to be displayed). If **SegmentsNo** is 0, than a big arrow is displayed in the display area (**TNAV fig1**). If not, the current crossroad is being displayed (**TNAV fig2**). Whereas, if, in the data area, the **SegmentsNo** is 0, the following information is displayed: the distance to the destination, the estimated time to the destination, the distance to the next significant crossroad, the time estimated to the next significant crossroad. If not - that is: if **SegmentsNo** is not 0, the information that is displayed is: the name of the current street, the distance to the crossroad and the name of the next street on which the car will be.

Figures 1 and 2 exemplify the behavior of TNAV in the situations specified above.

Fig.1 -- TNAV go ahead.

Fig.2 -- TNAV current crossroad.

TNAV control has some properties that determine what all it will display. There are two significant properties, namely **DisplayThreshold** and **NotifyThreshold. DisplayThreshold** and **NotifyThreshold** determine how many segments there will be displayed.

To display the route, we use the algorithms given below.

The numbers of segments, which must be displayed before the turning point, is calculated by making a sum of times. The first term of the sum is the time of the current segment, that is: the segment with the turn. Then we take the time of the previous segment and add it to the sum, and so on, until the sum is bigger than, or equal to, the DisplayThreshold value. If the sum is bigger then the DisplayTreshold value, we must display only a portion of the last segment [3].

To determine the segments that must be displayed after the turning point TNAV, we use the following algorithm. A sum of segments length is made until the sum is lower than, or equal to, the DisplayThreshold value. If these segments contain one or more turning points, then the segments will be displayed until the next segment to the last turning point. The last segment will be displayed with a length equal to a constant. Otherwise, it will be displayed a segment with a length equal to a constant [4].

For an optimal guide of the car, TNAV control has also a part of voice. Two sound events are implemented.

The first sound event specifies which turn the car must take. The algorithm used to determine this thing is the following.
The control calculates the angle between the current street and the next one.

The angle between the current street and the next one may be between –180 and +180.

If:

- abs(angle)<15 - the car should go ahead
- angle>15 - the car should turn left
- angle<-15 - the car should turn right.

After that, the program calculates how many streets there are before the turning point. For each branch of the crossroad before the turning point, the program translates the angle from [0…360] to [-180…180]. If the street is before the next street, then the program increments StreetNoS - if the street is on the left, StreetNoD - if the street is on the right.

When the algorithm ends, StreetNoD contains 1 + the number of streets that turn right before the next street, and StreetNoS contains 1 + the number of streets that turnleft before the next street. That number represents the number of the street in the crossroad. If the car should turn left, the event sends StreetNoS. If not, it sends StreetNoD.

The second sound event tells if there is a so-called proximity crossroad after the current one. The algorithm related to this sound event is the following. The control has a method used to set the distance to the turning point when the event is fired. Two crossroads are considered to be in proximity, if the distance between them is lower that a given distance. If there are two proximity crossroads the program calculates the value of the two angles, first from the first crossroad, the second form the second crossroad.If there is a single crossroad, that is: no proximity crossroad to the current one, the program calculates just the angle of the current crossroad.

To test the TNAV control we use a program developed in VisualBasic6.0. This is a program developed especially for this purpose. Two wrappers controls are used for the TNAV Control and PalmGeo Control. Their names are: **NavPlugIn** and **TestNavPlugIn**.
The test program functionality includes the functionality of the two basic controls TNAV Control and PalmGeo Control. To run this program, some data must be saved in the registries. These include: the path of the map file, the start GPS position, the town, the destination address (the street and the number). The following sequence must be followed, in order to let the program run:

- click the button **Set Long Lat Start** for setting the start position;
- click the button **GetTown** to set the destination town;
- click the button **GetStreet** to set the destination street;
- click the button **ComputeComplete** to compute the data needed to display the route;
- click the button **ShowAll** to display the TNAV Control and PalmGeo Control:
- click the button **StartTimer** to start the program itself.

If the user wants to stop the program, he can do this by pressing again the button **StartTimer** that becomes the **StopTimer**. The figure shows the test program.

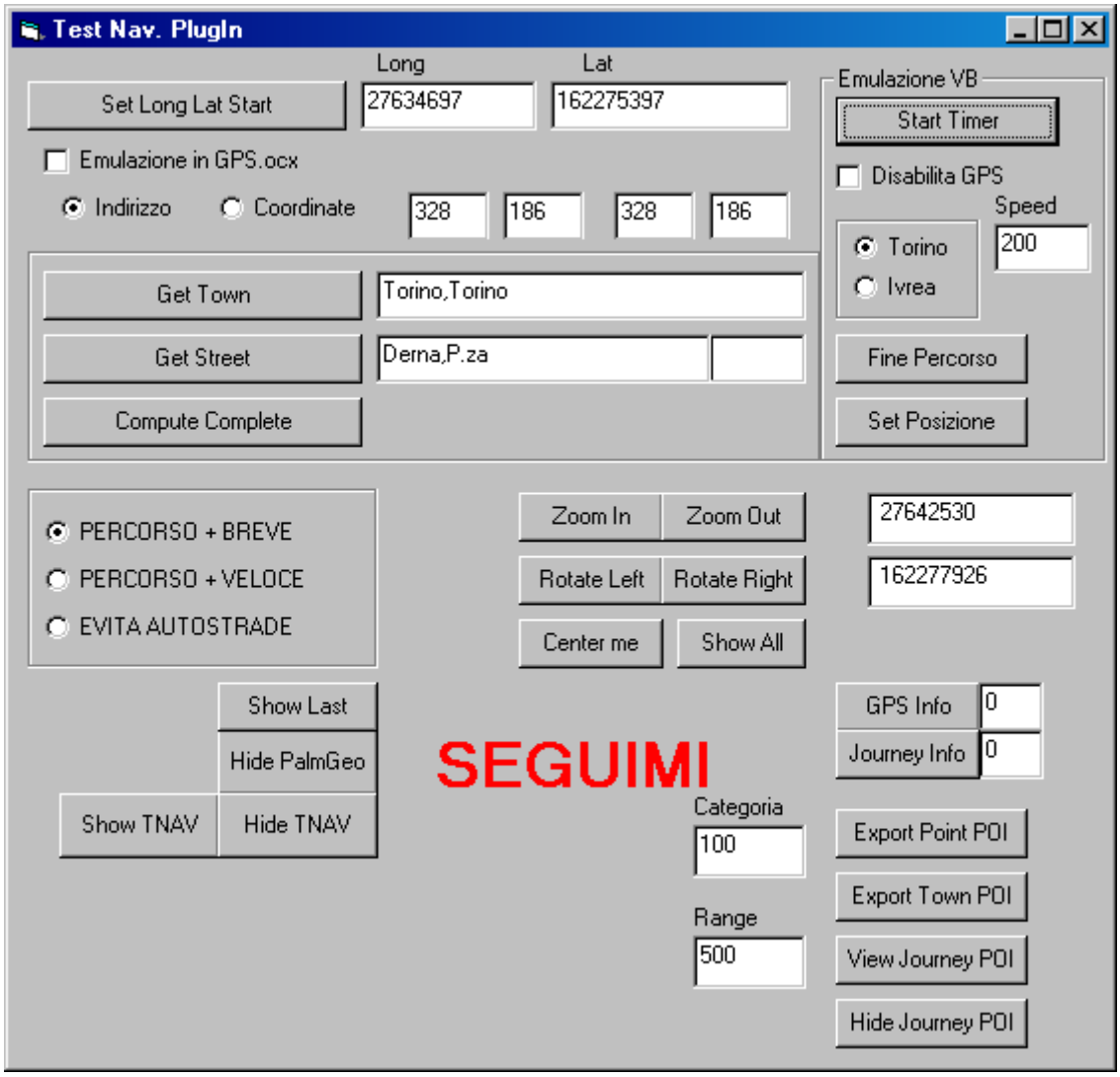

Fig.3 –Test program main form.

For a given position the PalmGeo Control displays the picture presented in **Fig.4**.

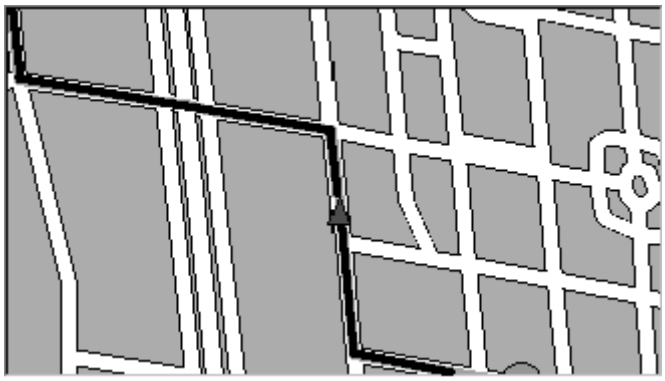

Fig.4 –PalmGeo the test program main form.

For the same position as in **Fig.4** the TNAV Control shows the picture presented in **Fig.5**

Fig.5 – TNAV current crossroad, for the same position as in Fig.4.

3. CONCLUSIONS

TNAV will simplify the navigation of cars and people. The car can be located even if an accident occurred. The application runs on PalmPC and communicates with a dedicate server using GSM system. Future enhancements will include the possibility of reading the town's map from a magnetic card and possibility to set the destination point (town, street) by vocal command.

BIBLIOGRAPHY

1. PalmGeo Control Documentation
2. TNAV Control Documentation
3. D.Gorgan "Elemente de Grafica" Editura U.T.C.N
4. Christiansen J., (1999), "MS Windows CE Graphics Features" [Grafica]