Spatial IT Application: Navigation System for Global Positioning System

Dr.G. Anil Kumar
Sr. Asst Prof CSE MGIT JNTU H TS India

Abstract

Market analysts estimate that location-based services will be a multibillion-dollar industry in the years to come. To obtain location parameters, a Global Positioning System (GPS) sensor is used, which retrieves the longitude, latitude and altitude of the sensors location. The longitude and latitude information can be mapped onto a digital map to display the position of a vehicle or a device, such as a PDA, in which the GPS sensor is embedded. Now a day, because GPS sensors are relatively inexpensive, they are being integrated into even mass-market items such as wristwatches. A case study of development of an embedded navigation system is intended to be implemented. The location information obtained from a GPS receiver is displayed on a digitized map. GPS simulator is also implemented in C as a learning experience. A navigation system consists of a GPS receiver interfaced to an embedded system. The GPS receiver calculates the location information (longitude. latitude and altitude) and sends it over a serial communication link to an embedded system. On the LCD screen of the embedded system, the map of the area we are traveling is shown; and on this map, the place we are will be highlighted. A GPS receiver can be directly interfaced to the embedded system. A GPS simulator in implemented in C language (for Linux platform), that generates the location information and also updates it every one second to simulate movement of a vehicle..

Keywords. *GPS, Navigation System, Spatial IT*

I. INTRODUCTION

To know where we are, is very important. We realize the importance of this information when we are lost in an unknown city or when we are hitch-hiking or when we are driving long distances. Navigation systems come in handy in such cases. In this mini project, a case study of the development of a navigation system is implemented as a learning experience. The location information obtained from a GPS receiver is displayed on a digitized map. GPS Simulator software is implemented for those who don't have access to a GPS receiver.

A navigation system consists of a GPS receiver interfaced to an embedded system. The GPS receiver calculates the location information (longitude, latitude and altitude) and sends it over a serial communication link to the embedded system. On the LCD screen of the embedded system, the map of the arear we are traveling is shown; and on this map, the place where we are will be highlighted. If we have GPS receiver, we can dirtectly interface it to the embedded system. Otherwise, we can write GPS receiver simulator software that generates the location information and also updates it every one second to simulate movement of a vehicle.

II. REVIEW OF LITERATURE

Serial Communications is one of the oldest mechanisms for devices to communicate with each other. Starting with the IBM PC and compatible computers, almost all computers are equipped with one or more serial ports and one parallel port. As the name implies, a serial port sends and receives data serially one bit at a time. In contrast, a parallel port sends and receives data eight bits at a time, using eight separate wires. Despite the comparatively slower transfer speed of serial ports over parallel ports, serial communications remains a popular connectivity option for devices because of its simplicity and cost-effectiveness. Although consumer products today are using USB connections, still a lot of devices use serial ports as their sole connection to the outside world.

Now a day, .NET Framework 2.0 and .NET Compact Framework 2.0 supports communication with other serial devices using SerialPort class. Applications for communication with a GPS receiver and extracting useful data for displaying the current location on a map are very important. GPS data can be piped to our PC via a serial port over Bluetooth and usage of this data to display a map using a mapping application, such as Microsoft Virtual Earth can be easily done.

A case study of development of an embedded navigation system is intended to be implemented. The location information obtained from a GPS receiver is displayed on a digitized map. GPS simulator is also implemented in C as a learning experience. A navigation system consists of a GPS receiver interfaced to an embedded system. The GPS receiver calculates the location information (longitude. latitude and altitude) and sends it over a serial communication link to an embedded system. On the LCD screen of the embedded system, the map of the area we are traveling is shown; and on this map, the place we are will be highlighted. A GPS receiver can be directly interfaced to the embedded system. A GPS simulator in implemented in C language (for Linux platform), that generates the location information and also

updates it every one second to simulate movement of a vehicle.

III. METHODOLOGY

A. *Development Environment:*

We need the following tools to develop the navigation system:

- GPS receiver such as Conexant's GPS receiver (www.conexant.com). This receiver has an RS232 port through which the location information is given out. We need to read this serial data and then decode the values of longitude, latitude, altitude and also the time in GMT.
- If we do not have access to a GPS receiver, simulator software is implemented to simulate a GPS receiver. This software runs on a Linux system. The simulator changes the location parameters every one second and transmits the data on the serial port to another desktop computer.
- A desktop computer running Windows Operating system receives the GPS simulator data from the Linux machine and shows the location on a digitized map.
- We need a digitized map on which we can show the position.
- We need a null modem cable to interconnect the Linux machine and the Windows machine.

Software used to develop the concept are Visual C++ 6, C, Linux. Infact, the requirement of GPS receiver (ex. Conexant's with RS 232 port), a digitized map are considered as hardware requirement.

B. *GPS Receiver Packet Format*

The GPS receiver sends the location information through the serial port in the form of a packet. As we can see from this format, gthe GPS receiver sends lot of information in addition to the longuitude, latitude, altitude and time. These parameters include the longitude and latitude directions, the number of satellites visible at that location, quality of the received signal etc. However inthis project, we are interested in the longitude and latitude parameters. The packet format is as follows:
$GPGGA, 222435, 3339, 7334, N, 11751, 7598, W, 2, 06, 1.33, 27, 0, M, -34. 4, M, 7, 0000*41

The various fields in this packet are:

Field	Field Type	Example
Start of the Packet		$GPGGA
Time	Hhmmss	222435
Latitude	xxxx.xxxx	3333.7334
Latitude direction	A	N
Longitude	Yyyyy.yy	11751.7598
Longitude direction	A	W
GPS Quality Indicator	X	2
Number of satellites in use	Xx	06
Horizontal dilution in precision	X.x	1.33
Altitude	x.x	27.0
Units of Altitude	M	M
Geodetic separation	x.x	-34.4
Units of Geodetic separation	M	M
Age of differential GPS data	x.x	7
Diffrential reference station ID	Xxxx	0000
Checksum	*hh	*41

Table 1.1 GPS Reciever Packet Format

The information obtained from the GPS receiver's serial port is decoded using this format. If we do not have a GPS receiver, the program (GPSSimulator.c) simulates the GPS receiver by generating the packet exactly in the above format. To simulate a moving vehicle, the longitude and latitude fields are changed every second and sent over the RS 232 link.

C. *GPS Simulator.C*

GPS Simulator.c runs on the Linux system. It simulates the GPS receiver.

How it Works?

- Open_port (void) function definition opens the serial port. /dev/ttyS0 is the serial port device file. For our Linux system, we need to use the appropriate filename as per the entry in the /dev directory.
- Config_port (void) function definition configures the serial port parameters. The data rate, number of data bits, number of stop bits and parity are set here.
- write_data (int fd, char *c) function is to write data top the serial port.
- SendSignals(int fd) is the function that simulates the GPS receiver and also keeps changing the values of longitude and latitude. The simulated data is stored in an array to send it over the serial link.

- The simulator uses an infinite loop to continuously generate the data.
- It obtains the system time.
- Latitude is changed by 0.03 to simulate movement.
- Longitude is changed by 0.03 to simulate the movement.
- Call to write_data (fd, array) function to write the array to serial port.
- It finally slleps for one second.

- Close _ port (int fd) call closes the serial port.
- Main (void) function opens the port, configures the port parameters, calls the SendSignals function and then closes the port.

We can compile the program usinf the following command:

$ gcc –O GPSSimulator GPSSimulator.c

We can execute the program using the command:

$ ./GPSSimulator

On the windows machine, we need to develop the navigation software that obtains the GPS data and shows the location on a digitized map.

6.6 GPSDlg.cpp

The file GPS Dlg.cpp contains the following method that reads the data from the serial port and decodes the location parameters.

How it works?

- It first loads the digitized map BITMAP3.bmp.
- Opens the COM2 serial port with bit rate 9600 bps, parity none, data bits 8 and 1 stop bit. If we are using a different COM port, we need to change it appropriately.
- The receiver uses an infinite loop to accept the data every second.
- Reads the data from the serial port byte by byte and check for the start of packet.
- Assigning the values of longitude and latitude direction to the latitude array.
- Assiging the values of longitude and longitude direction to the longitude array.
- Converts the latitude value from string to floating point number.
- Converts the longitude value from string to floating point number.
- Longitude is on X-axis and origin is 11751.7598
- Latitude is on Y-axis and origin is 3333.7334
- Sleep for 1.1 second as Linux machine sends the data every one second and we need to read that data and then process it.
- Plots the location on the map.
- Scales the increment. The variations in longitude and latitude are very small, and hence we scale it up to one pixel.

We need to invoke this method using the following code in GPSDlg.cpp:

```
CWinThread *m_RThread;
M_RThread = AfxBeginThread(Reciever, this);
```

IV. RESULTS AND DISCUSSIONS

Executing the Program

When we execute the GPS Simulator software on the Linux machine and the navigation software on the Windows machine, we can see the screen shot shown in the following figure 1.1:

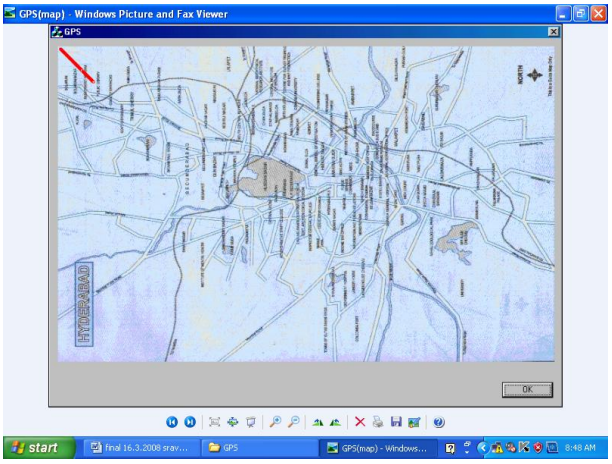

Fig. 1.1: Display the Location on the Digitized Map

On the top left corner, a thick line can be seen which shows the movement of the vehicle. If we have a GPS receiver, we can interface it to a laptop and run the navigation software on the laptop. Alternatively, we can port the navigation software on to an Embedded NT or Windows XP embedded system without any changes in the source code implemented. However, we need to obtain an accurate digitized map for our city/town. For most of the cities, accurate maps are not available. We can create a digitized map for our city ourselves by carrying a laptop fitted with a GPS receiver. Take a city bus, and at each bus stop, we note down the longitude and latitude. Then map these values on to the actual map – and we have the map for navigation.

V. CONCLUSION

In this mini-project implementation of a navigation system using a GPS receiver is done. The GPS receiver sends location information that contains longitude and latitude through an RS 232 port. This data is sent in the form of a packet with predefined format for each fieled. The navigation software has to decode the packet and obtain the longitude and latitude, and then indicates the position on a digitized map. The software that simulates a GPS receiver is also implemented. This software generates the packet as per the format of the GPS receiver and keeps changing the longitude and latitude values every second to simulate a moving vehicle. A Windows machine recieves the information through its RS232 port and displays the location on the map. The implementation details of the mapping software is also

given. The software written on the Windows machine can be ported on to the Embedded NT or Windows XP embedded platform to create an embedded navigation system

This application can be ported to the Embedded NT platform. However, the following files are required to run the application at the target i.e. GPS.exe application file, MFC42D.DLL, MFC042D.DLL, MSVCRTD.DLL. Create a component using Component Designer. Create the target design using the Target Designer.

Now a day, .NET Framework 2.0 and .NET Compact Framework 2.0 supports communication with other serial devices using SerialPort class. Applications for communication with a GPS receiver and extracting useful data for displaying the current location on a map are very important. GPS data can be piped to our PC via a serial port over Bluetooth and usage of this data to display a map using a mapping application, such as Microsoft Virtual Earth can be easily done

REFERENCES

[1] Essentials of GPS second edition, N.K.Agarwal Geodesy and GPS publishers, Hyderabad 2006.
[2] Data Communications and Computer Networks, Dr. K.V.R.K. Prasad, Dreamtech publishers 2003.
[3] Visual C++6 Complete Reference pappas et. al. Mc.grawhill 1999.
[4] http://www.trimble.com/gps/why.html
[5] http://www.cursor-system.com
[6] http://local.live.com
[7] http://dev.virtualearth.net/mapcontrol/v3/mapcontrol.js.
[8] A.M.Chandra and S.K.Ghosh, “Remote Sensing and Geographical Information Systems”, Narosa publications, India, 2006.
[9] Barrett E.C. and Curtis L.F., “Introduction to Environmental Remote Sensing”, Chapman and Hall, second edition 1982.
[10] http://www.gisdevelopment.net
[11] http://www.nrsa.gov.in.
[12] http://www.isprs.org/istanbul2004/comm2/papers/220.pdf