Chapter 11

The Design and Realization of a Portable Positioning System

Xin Xu, Kai Zhang, Haidong Fu, Shunxin Li, Yimin Qiu and Xiaofeng Wang

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/50483

1. Introduction

With the rapid development of positioning techniques, the design and realization of a positioning system is inclined to introduce user-friendly platform which can provide easy way to assist end users to locate themselves by utilizing either sensor-based or satellite-based measurements [4].

In the early 1960s, the Inertial Navigation System (INS) was developed by Litton Industries. INS is a sensor-based, self-contained, dead-reckoning positioning system which uses a computer, motion sensors and rotation sensors to perform continuous positioning. The computation of position, velocity and attitude information of moving objects in INS is different from GPS-based methods. This information is calculated by an inertial measurement unit (IMU), where the difference relative to a known starting position, velocity and attitude can be obtained [20]. However, INS suffers from integration drift and leads to unbounded accumulation of errors when calculating the time varied information, due to the needed integrations of small errors in the measurement of acceleration and angular velocity [28], [18]. Therefore, standalone INS-based positioning is unsuitable for accurate positioning over an extended period of time.

Then, in the early 1970s, the US Department of Defense (DoD) created the first Global Positioning System (GPS). GPS is a typical example of space-based satellite navigation system that provides sufficiently accurate information including time, position and velocity [11]. However, GPS cannot work under specific conditions where there lacks an unobstructed line of sight to four or more GPS satellites [3], [29]. In the GPS-denied environments such as urban canyons and tunnels, it is difficult for GPS-based measurements to perform continuous and reliable positioning [15]. Due to the lack of line-of-sight between the receiver's antenna and the satellites in these circumstances, blockage, interference, jamming and multipath effects may easily influence the GPS to perform its function of locating the position of receiver [7].

As mentioned above, either GPS-based or sensor-based measurement cannot provide a continuous and reliable solution. An alternative solution can be obtained by integrating measurements from sensor-based systems with GPS-based measurements. The integration is reasonable and can improve the performance by mitigating each others' problems. As the characteristics of sensor-based measurements are complementary with those of GPS-based positioning systems.

In the past, several previous efforts have been made to realize the integration of GPS-based and sensor-based positioning systems. In [12], the GPS was integrated with a full IMU, where three accelerometers and three gyroscopes were contained. In [23], a reduced inertial sensor system (RISS) was proposed, which integrated the GPS with only one gyroscope and an odometer or wheel encoders, to provide positioning information. In [24], Abdelfatah realized this RISS and GPS integrated positioning system. The developed system can compute the data-fused positioning on a field programmable gate arrays (FPGA).

Most of the recent positioning systems can provide effective and efficient solution. However, these products may face many difficulties like long development cycle, high development cost, and short life cycle. In addition, the size and weight of most positioning systems are not easy for end user to use and to carry. The design and realization of portable positioning systems (PPS) is important, which can provide easy way for consumers to locate themselves anywhere. Fortunately, the System On Programmable Chip (SOPC) technology can integrate various modules in a single FPGA, including CPU, memory storage, I/O interface, and etc. And this will provide a great flexibility in hardware configuration. Besides the requirement of PPS hardware, the requirement in PPS software is also need to be considered. The micromation and portability requirement in the design of PPS hardware will inevitably result in more limitation in the storage capacity and processing speed of PPS software. As a result, few storage space and rapid transmission of cartographical is needed in the design of PPS software to ensure the end user experience and runtime efficiency.

The goal of our work described in this chapter is to develop and implement a SOPC based PPS which is capable of handling the data-fused positioning and can provide user a feasible way to obtain their position anywhere.

2. Framework of PPS

The PPS generally includes two main components: receiving terminal and monitoring center. As illustrated in figure 1, the receiving terminal can receive the GPS signal from satellite, which is then fused with the signal from built-in digital compass to perform a combined positioning. Then, the resulting information is transferred to the monitoring center through wireless network and several base stations. In monitoring center, the received positioning information is analyzed and matched with high precise digital map to display the location of the receiving terminal in real time. Finally, the resulting location can be sent back with the compressed digital map to display the location of receiving terminal on its own screen.

In this chapter, we will introduce the design and realization of the PPS, in which three core contents are included.

1. The design and realization of receiving terminal (PPS hardware). A solution integrating measurements from GPS and digital compass is proposed to overcome the problems that arise from using GPS positioning systems in standalone mode. This method can provide

continuous and reliable positioning, even under the circumstances of urban canyons, tunnels, and other GPS-denied environments.

2. The design and realization of monitoring center (PPS software). The processing speed is important for the long-term stable operation of PPS. In order to improve runtime efficiency of the PPS, we compare the performance of native table file and memory table in MapX by using some frequently used operation. By calculating each response time, a suitable storage type can be found out.

3. Digital map compression for PPS. A new digital map compression algorithm is presented to provide low cost storage and rapid data transmission in PPS. This algorithm overcomes the problems that arise from the three traditional compressing algorithms, and can provide reliable and efficient performance.

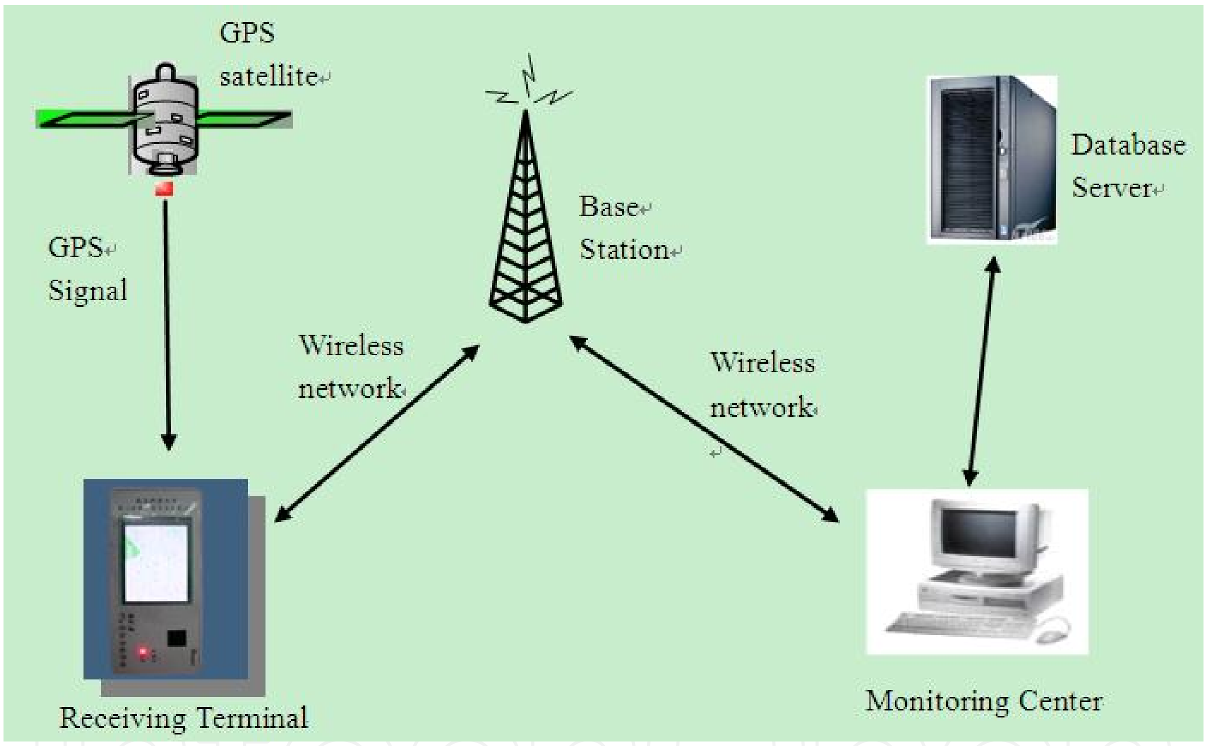

Figure 1. Framework of PPS.

3. Design and realization of receiving terminal

Several hardware platforms can be used for the development of receiving terminal, such as microcontrollers (μC), digital signal processors (DSP), field programmable gate arrays (FPGA) and application specific integrated circuits (ASIC) [16], [9], [22]. The choosing of one hardware platform over others should depend not only on the requirements of PPS such as the performance, power consumption, cost per chip; but also on the easy to use of the tools for producing PPS within the constrained system cost and project time.

The design and realization of receiving terminal in PPS uses FPGA as the development platform. The single FPGA can be integrated, using SOPC technology, with various modules including CPU, memory storage, I/O interface, and etc. And this will provide great flexibility in hardware configuration. The specific hardware description of receiving terminal is illustrated in figure 2.

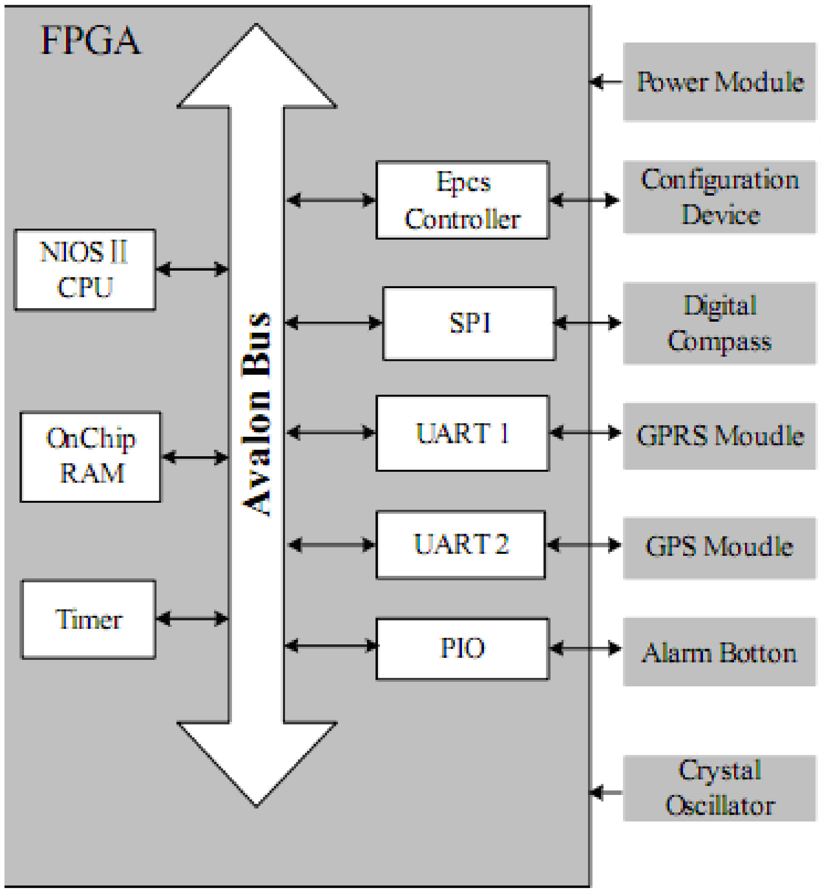

Figure 2. Framework of Receiving Terminal.

3.1. Soft core processor

The receiving terminal is implemented using soft-core processor. A soft-core processor is a microprocessor fully described usually in a Hardware Description Language (HDL), which can be synthesized in programmable hardware. Soft-core processors implemented in FPGAs can be customized easily according to the needs of a specific target application.

Nios II, developed by the Altera Corporation, is a 32-bit general-purpose Reduced Instruction Set Computing (RISC) processor-core [1]. Compared with traditional portable positioning system, Nios II embedded receiving terminal has many advantages:

1. High performance and low price. Nios II soft-core processor has full 32-bit instruction set, data path, and address space. It can perform beyond 150 DMIPS, but only costs 35 cents to implement in a low-cost Altera FPGA device [2]. Besides, the integration implementing of processors, peripherals, memory, and I/O interfaces on a single FPGA can also contribute to the reducing in cost.

2. High flexibility in design. SOPC development tools enable developers to customize the exact peripherals, memory storage, and interface component freely. This is somewhat similar to the component software development, developers can download and configure the needed components in a single FPGA according to different applications.

3. Easy-to-update. By implementing a soft-core processor in a FPGA, in-field hardware component can be updated conveniently. As a result, new features and latest standards can be incorporated easily.

3.2. FPGA

As illustrated in figure 2, the GPS and digital compass modules were both connected with I/O interfaces in FPGA. The signal from satellite can be received by the GPS module, and then fused with the signal from digital compass to perform integrated positioning. In the receiving terminal, GPS module took use of GSU-40 developed by Nihon Kohden Corporation. The size of GSU-40 is 26*26*97 mm3, which meets the portability requirement of the PPS; besides, the positioning data is updated every second to ensure accurate positioning over an extended period of time. The data format in GSU-40 complies with the NMEA-0183 standard.

NIOS II Soft-core CPU is connected with other IP cores via Avalon on-chip bus, which defined time sequence of port and communication between master units and slave units. The hardware design can be accomplished conveniently by using the SOPC Builder system development tools. According to function requirement in this system and high flexibility of NIOS II Soft-core processor, IP cores are designed in FPGA.

Besides, other components were also integrated in this ALTERA CY1C12Q240C8 based on SOPC technology, including NIOS II Soft-core processor, On-Chip RAM, Timer, digital interface and etc. NIOS II/e "economy" soft-core processor: Three kinds of optional processor modes are provided by NIOS II, in this system, NIOS II/e "economy" core is employed which is designed to achieve the simplest possible core at the smallest possible size; Serial Peripheral Interface (SPI): A communication interface used to obtain the positioning data from digital compass; Universal Asynchronous Receiver Transmitter (UART): The interfaces for GPRS module and GPS module to communicate with CPU; On Chip RAM: CY1C12Q240C8 FPGA provides as many RAM memories as 239,616 bits, which are enough for the requirement of PPS hardware system; Parallel Input/Output (PIO) module is used as alarming module, which enable the connection with CPU via PIO to send alarming message to the monitoring center; Epcs Controller module is used to control hardware configuration files and download programs from serial configuration device to FPGA when the system is power up; Timer module is used to provide system time interruption; Power module uses Lithium battery and provide power to PPS hardware system; Crystal Oscillator module provides system clock to PPS hardware system; and An EPCS4 chip is used to store the configuration and software data.

3.3. Digital compass

The signal of digital compass is not from the traditional magnetic needle compass or the gyrocompass, but from an electronic solution (3-axis magnetic sensor and tilt sensor) which adopts the principle of earth magnetic field. The 3-axis magnetic sensor collects the 3-axis earth magnetic field signal; while the tilt sensor is used for compensating "roll" and "pitch". This electronic solution can provide the advantages of a solid-state component without moving the parts and the convenience to interconnect with other electronic systems [6]. The structure of digital compass is shown in figure 3.

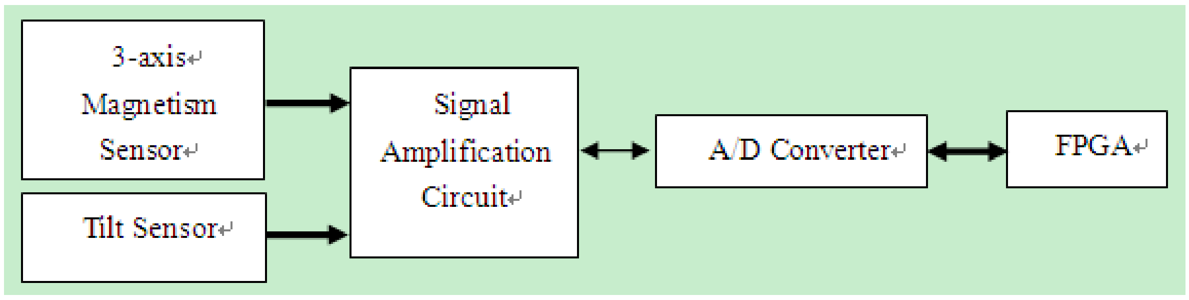

Figure 3. The structure of digital compass.

3-axis magnetic sensor collects the 3-axis earth magnetic field signal and the tilt sensor is used for compensating "roll" and "pitch", and then all of them are amplified and convened to digital signals through A/D converter, at last transmitted to NIOS II by PIO interface.

3.4. Integrating positioning algorithm

GPS positioning signal may be blocked in underground park, tunnel or other GPS-denied places. To solve the problem, this system adopts digital compass to realize combined positioning. An adaptive dead reckoning algorithm is used for fusing the two positioning information.

Dead reckoning is a typical positioning technology in the two-dimensional plane and widely used in vehicle positioning. Once the start location (x_0, y_0) and azimuth (θ_0) angle are given, the current location could be calculated from the prior traveled distance (s_i) and shift angle $(\Delta\theta_i)$. Equation (1)-(3) gives the formula for dead reckoning:

$$Lat_{compass} = x_k = x_0 + \sum_{i=0}^{k-1} (s_i * sin\theta_i) \tag{1}$$

$$Lon_{compass} = y_0 + \sum_{i=0}^{k-1} (s_i * cos\theta_i) \tag{2}$$

$$\theta_i = \theta_{i-1} + \Delta\theta_{i-1} \tag{3}$$

Dead reckoning position error is accumulating with the procedure continues because the current calculated position during each sampling period depends on the previous calculation cycle. So in PPS, we adopts the adaptive dead reckoning algorithm proposed in [21] to improve the precise of this portable position system. The azimuth information is collected by digital compass. When the PPS located not in GPS-denied environment, the two position devices are both used to calculate the location, not only the DR system status are amended constantly, but also exact start location and azimuth information can be provided by the combined positioning output.

The characteristics of INS-based measurements are complementary with those of GPS-based positioning systems. The positioning information from GPS can be used as the initial value and the error correction of digital compass; on the other hand, the signal from digital compass can be used to compensate the random error in GPS to provide accurate positioning, and also can smooth the positioning trajectory. Thus, the resulting positioning information can be given by the following formula:

$$Latitude = (1 - \beta) * Lat_{compass} + \beta * Lat_{GPS} \tag{4}$$

$$Longitude = (1 - \beta) * Lon_{compass} + \beta * Lon_{GPS} \tag{5}$$

Latitude in (4) and Longitude in (5) are the results of positioning coordinates. $Lat_{compass}$ and $Lon_{compass}$ are the coordinates of digital compass; while Lat_{GPS} and Lon_{GPS} are the coordinates of GPS. β is the weight value and varies according to different positioning environments, which can be modeled by fitting the empirical data.

3.5. Experimental results

Figure 4 and figure 5 are the tracking results of the two positioning modes in the monitoring center. As illustrated in figure 4, the standalone GPS based method fails to perform its function in locating its position when it is in the GPS-denied environment (urban tunnel). However, this problem was successfully tackled using the integrated GPS/INS based positioning. The continuous positioning trajectory can be found in figure 5.

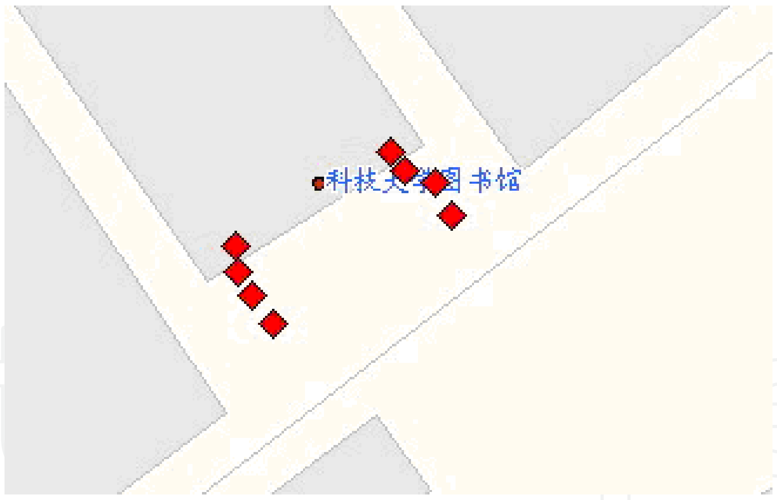

Figure 4. Result of standalone position solution.

By comparing these two tracks of the same object moving on the same route, we can find that standalone GPS based positioning fail to work in the GPS-denied environment regularly, where the line-of-sight between the receiver's antenna and the satellites is lacked. On the contrary, the proposed integrating positioning solution which adopts the positioning information both from GPS and digital compass works well, which can provide continuous and reliable positioning solution under the circumstances of urban tunnels.

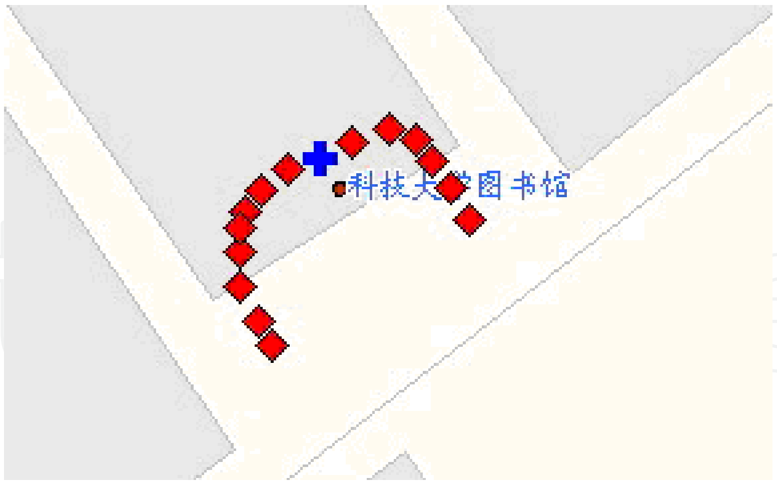

Figure 5. Result of integrated position solution.

Besides, we also conducted a number of testing in different GPS-denied environment. All of them indicate that a good positioning effect can be gained by adopting this integrating positioning technology, which provides a feasible way for the development of the PPS.

4. Design and realization of monitoring center

Real-time monitoring of multiple targets is crucial in PPS. However, due to the frequent status change of each monitoring target, large amount of data was generated during the whole runtime of system. As a result, longer system response time will be taken to process these data and display them on the digital map. In this circumstances, it may incur a problem that follow-up data may have to wait until previous points have been processed. This will worse the runtime efficiency of system. In our monitoring center, MapX component is used for the processing and displaying of digital map. MapX provides a variety of object, properties and methods, among which the Animation Layer can be used as an efficient mean for real time monitoring. However, the main objective of Animation Layer aims to avoid the vision fluttering bring by the fast refresh rate, which is different from our goal and cannot effectively solve this problem.

Conventional approach for reducing the response time of system can be divided into two categories. The first method is to extend the time interval between the receiving of locating information, and realize the real-time positioning through the use of some technology, such as dead reckoning etc, which can assist for positioning. However, it is not a reliable and continuous positioning solution. Another method for real-time positioning is to optimize the runtime efficiency of system by improving the speed of information processing. Specifically, we use memory table instead of native table file to process the locating information, which can avoid frequent I/O process, thereby improve efficiency of the system. In order to illustrate the effectiveness of this approach, we respectively use native table file and memory table in MapX to add and modify the map objects. Then a comparison is made to find the difference in efficiency of the system by calculating their response time.

4.1. Testing methods selection

The MapX Object Model is hierarchical tree structure. Every object, properties and methods in MapX are derived from the Map Object, which located in the top-level of the tree. Maps are the basic building blocks for MapInfo MapX. Each map is defined by three objects: Layer, Dataset and Annotations objects and collections [19].

The first important object is Layer Object and Layers Collection.

Each Map has a collection of layers. The Layers collection is made up of Layer objects. The Layers collection has methods and properties used to add and remove Layer objects from the collection. Computer maps are organized into layers. Think of the layers as transparencies that are stacked on top of one another. Each layer contains different aspects of the entire map. Each layer contains different map objects, such as regions, points, lines and text. For example, one layer may contain state boundaries, a second layer may have symbols that represent capitals, a third layer might consist of text labels. By stacking these layers one on top of the other, you begin to build a complete map. You can display multiple layers at a time. Map layers form the building blocks of maps in MapX. Once you have created your map of layers, you can customize the layers in a variety of ways, add and delete layers, or reorder them.

Map layers display in a particular order. It is important to order your layers correctly. For example, you have a layer of customer points and a layer of census tracts. If the layers are incorrectly ordered in the map window, MapX might draw the customer points first and then display the census tract layer second. Your points would be obscured beneath the census tract layer.

Another object is Dataset object and Datasets collection.

Each Map has a collection of Datasets. The Datasets collection has methods and properties used to add and remove Dataset objects from the collection. Datasets enable you to bind user data to your maps. For example, if you have a Microsoft access database of sales by county and you had a Lotus Notes database of the location of your sales force, you could bind that data to a map and spot trends or notice correlations between the two sets of data. There are many different types of databases in businesses today; therefore, MapX lets you bind to several different types of Data Sources. In MapX, the data is represented as a Dataset object.

The last one is Annotation Object and Annotations Collection.

Each Map has a collection of Annotations (Map.Annotations property). Annotations are either symbol or text objects, and are drawn on top of the map. Annotations are typically used to add messages (text) to a map or to put symbols on a map. These annotations will scale with the map as you zoom in and out. They are not necessarily associated with a particular layer of the map and are always on top.

The greatest impact on the runtime efficiency of system lies in the operation which processes and displays large amount of data to the digital map. According to the preceding discussion on the MapX Object Model, we can confirm that this operation mainly involved in two objects: Layer and Annotations objects and collections. The Dataset object and Datasets collection does not seriously affect this operation. Therefore, we decide to use AddFeature, AddAnnotation and Update these three methods for testing.

4.2. Experimental results

The efficiency of the system can be measured by their response time at runtime. The less response time system needed the higher efficiency system worked in. On the other hand, the more response time system needed, that means longer system response time needed to be taken to process the locating information, and therefore took a toll on efficiency of system. In order to evaluate the efficiency of system, we respectively used native table file and memory table in MapX to add and modify the map objects. Then their response time was calculated. Firstly, we created a temporary layer for testing:

layerinfo.Type = miLayerInfoTypeTemp

Then the storage type, TableStorageType in MapX, of this layer was set up to native table file and memory table using the AddParameter method. For example, we used the following command to set the storage type to native table file:

layerinfo.AddParameter "TableStorageType", "Native"

After the AddFeature, AddAnnotation and Update methods were used into this layer, their respective response time was calculated.

Specifically, we at first calculated the response time needed by AddFeature method to add M*N nodes. And the Timer component was used in telling time. Calculating the response time of AddAnnotation method is similar to AddFeature.

Then the response time can be calculated by using the Update method to operate the M*N nodes. Likewise, the Timer component was used in telling time. And after these operations, their respective response time to memory table and native table file was calculated.

The experiment result was shown in table 1. AddFeature, AddAnnotation and Update in the first line of table corresponded to the operation of adding nodes, adding annotations and updating nodes status. The mem and file in the second line of table corresponded to memory table and native table file. The Nodes in row 1 stood for the number of nodes. The data in table 1 was the result of response time for corresponding operation on the nodes. Let us took the first data in table 1 for example. The number 0.1725 stood for the response time when adding 900 nodes into the memory table.

The experimental results indicate that the use of memory table could avoid frequent I/O process, thereby improved the efficiency of system. Due to large amount of data brought by the frequent status change of each monitoring target, overlong response time would result in the deteriorating of system runtime efficiency. Accordingly, it is suitable to use memory table as storage type in personal portable positioning system. This result would be helpful for other research in which the status of monitoring target needed to change frequently, such as industrial valve monitoring, telecommunications terminal monitoring etc.

5. Digital map compression for PPS

Low cost storage and rapid cartographic transmission is important for PPS. And whether or not the digital map can be entitled low cost storage and rapid transmission is subjected to the efficient compression of the digital map [31].

	AddFeature	AddAnnotation	Update
30*30 mem	0.1725	0.09375	0.28125
30*30 file	7.687625	0.09375	9.313
40*40 mem	0.344375	0.203125	0.593875
40*40 file	14.98094	0.203125	17.43406
50*50 mem	0.6095	0.40625	1.062875
50*50 file	24.65269	0.421875	27.94931
60*60 mem	1.031875	0.75	1.8445
60*60 file	36.87169	0.765625	40.98056
70*70 mem	1.844375	1.40625	3.266
70*70 file	51.12194	1.46875	55.93431

Table 1. Comparison of AddFeature, AddAnnotation, and Update methods on memory table (mem) and native table (file)

Digital map can be divided into two categories: vector structures and grid structure [26]. And they have their own advantages and disadvantages in expressing different geographical phenomenon. The compressing of grid map is not involved in this paper, for it is similar to image compression. On the other hand, vector data compression can be simply defined as the process of eliminating redundancy. As stated by ZHENG [30], vector data compression was extracting a subset from data sets with possibly little data, and at the same time reflecting the original appearance of it as far as possible. Depending on the nature of the application, two classes of compression methods exist: lossless methods and lossy methods. In the lossless methods, data is compressed in such a way that upon reconstruction (decompression), the original data is exactly restored. In the lossy methods, an error measure between the original and the reconstructed data is used to allow a tolerable data distortion. Most vector map compression techniques, including the algorithm introduced in this paper, are of this last category, and eventually a combination of the two categories. Structural tables, namely geometry data, were usually used to store the graphical and topological information of vector map in current applications of GIS, such as MapInfo data. Other data, attribute data for example, was saved in other files. Moreover, attribute data could be saved in files or database, while geometry data was saved in files in most of the time. Still, it organized the files through different layer according to their geographical information. Generally, geometry data, compared to attribute data, took the most of the vector map's storage. For instance, MapInfo vector map's layer, which was saved in file format, generally had two textual file, namely *.MAP for geometry data's storage and *.TAB for attribute data's storage. And this paper will focus on the compression of the *.MAP file.

Specifically, vector map compression is based on the principle of selecting a reduced set of dominant points (DPs) on the curve from the map according to an adequate strategy. Currently, there are many traditional algorithms for compressing vector data. However, they have their own problems when applying the compression to vector map. As a result, an enhanced vector data compressing algorithm was put forward in this work, whereafter, the reliability of this algorithm was tested through experiment. The result of the experiment shows that the enhanced vector data compressing algorithm introduced in this paper could efficiently tackle the problems caused by the classical compressing algorithms, especially effective when aiming at the vector map. Besides, it will hopefully serve as useful feedback information for related works.

5.1. Traditional compressing algorithms for digital map

Three types of geographic information have been developed to represent the vector map over the past years, namely point element, curve element and surface element [13]. Point element, similar to point in geometric feature, stands for location of the target in ground. And two-dimensional coordinates was used to express its spatial information. For instance, we can use point element to express hospital, station, pier, shop and so on. Curve element, similar to curve in geometric feature, can stand for road, river, railway and etc. Surface element, similar to polygon or closed region in geometric feature, usually has the character of area and perimeter. For example, we can use surface element to express farmland, grassland, Public Park, Public Square and so on. Among them, compression towards point element will not have significant effect in saving storage space. Surface element compression can be seen as the compression towards a family of curves [27]. Therefore, compressing towards vector map can be treated as the compressing towards curve element, namely curve simplification.

5.2. Definitions of curve simplification

Curve simplification [5]: Given a polygonal curve $C_{A,B} = (P_1 = A, P_2, ..., P_N = B) = P$, where A and B are the curve endpoints and N is the number of samples, the curve simplification of $C_{A,B}$ consists in computing another polygonal curve $C'_{A,B} = P'(A = Q_1, ..., Q_M = B) \in C_{A,B}$, with $M < N$ and $[P, P'] < \epsilon$, with $\epsilon > 0$, a preset tolerable error of simplification.

5.3. Limitation of traditional curve simplification algorithms

Traditional compressing algorithms can be successfully applied to vector data. However, they had their own problems when applying the compression to vector map.

In James Algorithm [32], due to its' lack of consideration of the curves' global features, the overall direction of the curve may be distorted when there was a continuous small angle change. As shown in figure 6, we calculated the angle change α_2 between line $L_{1,2}$ and line $L_{2,3}$ through the point 2. Then the angles change α_2 was less than the tolerance band α_0. As a result, the point 2 should not be preserved. And similar statements could be made about the points 3, 4, 5 and 6. At last the curve $C_{1,2,3,4,5,6,7}$ could be simplified to line $L_{1,7}$ with a distortion of the overall direction.

Douglas - Peucker Algorithm [8], which is the most popular method to reduce the number of vertices in a digital curve, can keep the overall direction well. The algorithm iteratively selects new points for inclusion in the thinned output curve based on their deviation from a baseline connecting two neighboring points already chosen for inclusion. A review of relevant GIS literature, in journals and online, indicates that implementations often incorporate an error in the method used to calculate the distance between baselines and intermediate data points.

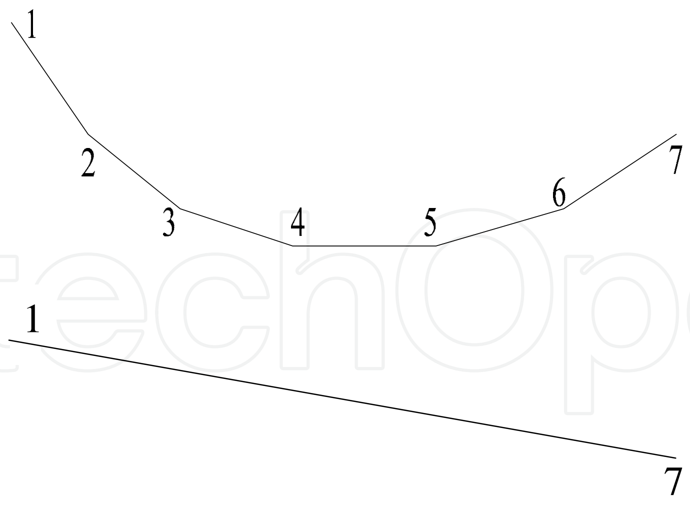

Figure 6. Distortion of the overall direction.

This problem exists because the original Douglas-Peucker paper is somewhat ambiguous in its definition of the distance criterion for selection of a point [8]. On pages 116 and 117 of their paper we see: (1) The perpendicular distance of C from the segment A-B, (2) distance from the straight line segment, (3) the furthest point from the straight segment, (4) maximum perpendicular distance from the segment, (5) the greatest perpendicular distance between it and the straight line defined by the anchor and the floater. Of these definitions, (2) and (3) are correct, but (5) seems to be the most widely used by programmers [10]. We need the distance from the segment, not the distance from the line or the perpendicular distance from the segment. To see the effect of this mistake, consider the situation shown in figure 7. When the curve witnessed a significant angle change, some characteristic bending points (point P in figure 7) might be lost after compression and resulted in terrain distortion.

Light Column Algorithm calculated through ever-dwindling caliber to detect the point outside the fan-shaped region [14]. Owing to the increasingly stringent conditions of detection, the compressing was not that efficient. Still, similar to Douglas - Peucker Algorithm, when the curve saw a significant angle change, some bending feature points might be lost and resulted in terrain distortion. A simpler example, which could be used to test whether an implementation of the Light Column Algorithm suffers from this mistake, may be seen in figure 8. The modified line segment will miss the removed point during the line $L_{B,P}$ and $L_{P,A}$.

In spite of the reduction in tolerance band could solve the above problems; however there, at the same time, will inevitably be a significant reduction in the compressing ratio. Accordingly, some improvement on compressing algorithm for vector map was investigated in this paper.

5.4. Proposed enhanced vector data compressing algorithm

In order to solve the limitation of classical algorithms and extract as less data as possible without reduction in accuracy, it was considerable to integrate the merits of the above classical algorithms.

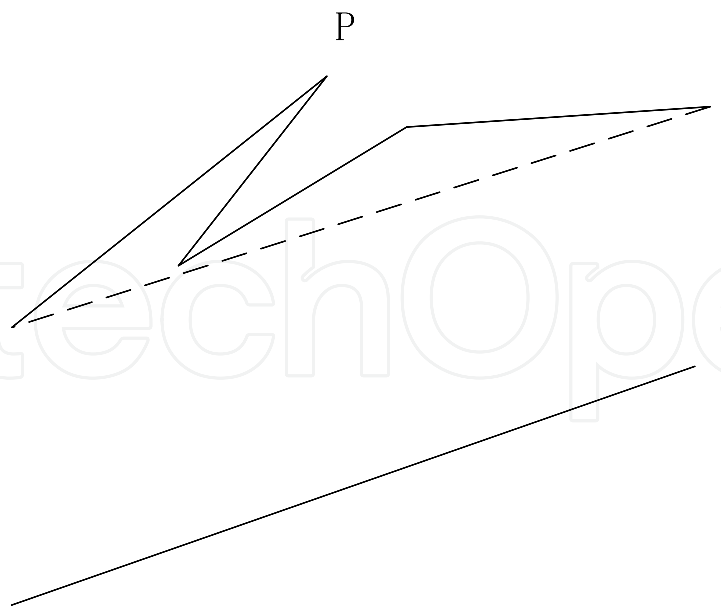

Figure 7. Terrain distortion.

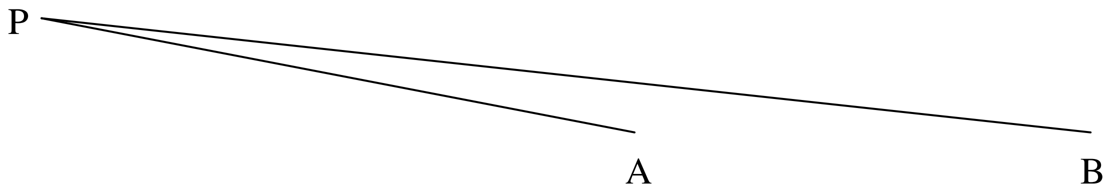

Figure 8. Polyline with three vertices.

Procedure of this algorithm:

(1) Using James Algorithm to extract the characteristic point with a significant angle change, and compressing the curves by Light Column Algorithm at the same time.

For all points in each curve, take P_i for example, the angle change α_i between line L_{P_{i-1},P_i} and line $L_{P_i,P_{i+1}}$ was calculated. Afterward, whether or not the point P_i witnessed a significant angle change and should be preserved was subjected to whether or not the angle change α_i was more than the tolerance band α_0.

On the other hand, we drew a straight line perpendicular to the line L_{P_{i-1},P_i}, which intersect the edge of the fan-shaped region defined by Light Column Algorithm at b_1 and b_2. Then, a_1 and a_2, which was located in this straight line and d/2 apart from the point P_i, was connected separately to P_{i-1} for defining a new sector. Besides, if the variable flag was equal to the true value, a_1 or a_2 should be replaced by b_1 or b_2 when they were outside the sector. As a result,

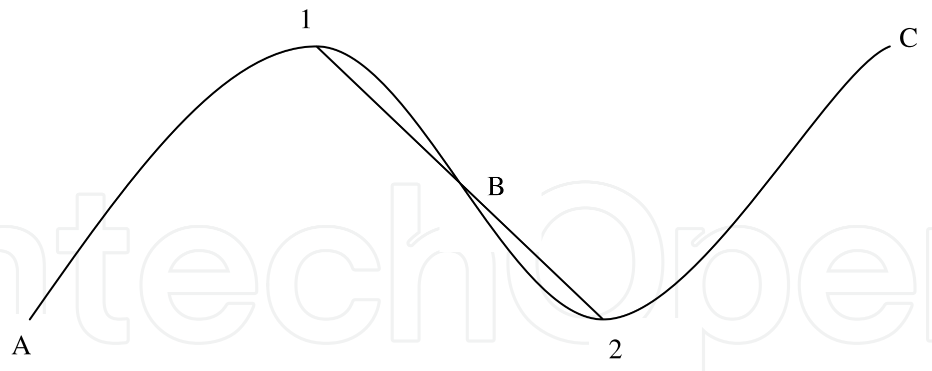

Figure 9. Endpoint redundancy.

the sector was updated. Afterward, if P_{i+1} was inside this sector, P_i should be replaced by P_{i-1}, and set flag to true. Else flag should be set to false.

Then the first step of compression was completed, and the point with a significant angle change was marked at the same time.

(2) Cutting the compression curve through the point with a significant angle change, and then compressing them separately by Douglas - Peucker Algorithm.

There were some problems, which had been discussed in reference [27], should be pay attention to in using Douglas - Peucker Algorithm to compress the vector map.

Closed curve should be divided into two curve end to end, and then compressing them separately by Douglas - Peucker Algorithm. The principle for selecting the split points lay in three criterions.

a The split point should be one of the DPs.
b The other split point should be the point with the greatest distance between it and the split point.
c The split curve should be the longest one of DPs' connection.

Redundant data may follow the compression to a family of curves linked end to end, for the initial point and the endpoint were extracted as the characteristic point for each curve. As shown in figure 9, point 1 and 2 respectively stand for the nearest vertices to endpoint B, therefore the curve $C_{1,B}$ and $C_{B,2}$ can respectively be replaced by subtense $L_{1,B}$ and $L_{B,2}$. However, the entire curve $C_{1,B,2}$ can be replaced by subtense $L_{1,2}$ with reduction of point B.

Accordingly, additional procedure is needed: detecting whether or not the endpoint (point B in figure 9) can be deleted.

a If the two succession curves were the public edge of a polygon, then the public point should be preserved.

b Else if the curve between the two points which were extracted from the two succession curves satisfied the criterion of Douglas - Peucker's tolerance band, then the joint point can be deleted.

c Else the point should be preserved.

In this program, depth-searching algorithm was adopted to find the public edge of the polygon [25].

After these two steps of compression above, the curve was compressed. Also, the procedure associated to this strategy was given in Algorithm 1, where $C_{A,B}$ is the curve needed to be simplified and J, G and D were the tolerance band of James, Light Column and Douglas-Peucker Algorithm. Besides, α_k stood for the angle change between line $L_{k-1,k}$ and line $L_{k,k+1}$ through the point P_k; LCA(P_i, P_k, G) stood for a sector defined by points P_i, P_k and tolerance band G; DP($C_{A,B}$, D) stood for procedure of the Douglas-Peucker Algorithm.

```
Algorithm 1: CurveSimp($C_{A,B}, J, G, D$)
Input: $J$, $G$ and $D$ were the tolerance band of James, Light Column and Douglas - Peucker Algorithm,
       $C_{A,B}$ is the curve needed to be simplified.
Output: $P'$ is the set of dominant points after compression, M is the number of the points.
1  $P' \leftarrow \emptyset \cup P_1$, $i \leftarrow 1$, $k \leftarrow 2$, $flag \leftarrow false$;
2  while $P_k \in C_{A,B}$ do
3      if $\alpha_k \geqslant J$ then
4          $P' \leftarrow P' \cup P_k$;
5      else
6          LCA($P_i$, $P_k$, G);
7          if $flag \neq false$ then
8              $a_1 \leftarrow b_1$, $a_2 \leftarrow b_2$;
9          end
10         if $P_{k+1} \in LCA(P_i, P_k, G)$ then
11             $P_k \leftarrow P_i$,$i \leftarrow i+1$,$flag \leftarrow true$;
12         else
13             $P' \leftarrow P' \cup P_k$, $flag \leftarrow false$;
14         end
15     end
16     $k \leftarrow k+1$;
17 end
18 if $P_1 == P_N$ then
19     CurveSimp($C_{A,C}, J, G, D$); CurveSimp($C_{C,B}, J, G, D$);
20 end
21 $M \leftarrow \mid P' \mid$;
22 DP($C_{A,B}$, D);
23 return ($P'$, M);
```

5.5. Performance evaluation

The reliability of algorithm can be tested through the comparison of the data, including the total length and average coordinate of the curve, with corresponding data before compression. The less difference in corresponding data, the higher fidelity was.

Name	Length (km)	Difference
Original curve	9.023876126834	0
James Algorithm	8.951782492573	7.98921E-3
Douglas-Peucker Algorithm	9.013681028737	1.12979E-3
Light Column Algorithm	9.015937028463	8.7979E-4
This Algorithm	9.016173945937	8.5353E-4

Table 2. Length Comparison

Name	Coordinates (km)	Difference
Original curve	114.385484755	0
	30.6358562641	0
James Algorithm	114.385866870	3.3406E-6
	30.6349333413	3.0126E-5
Douglas-Peucker Algorithm	114.385768397	2.4797E-6
	30.6353483255	1.6580E-5
Light Column Algorithm	114.385682864	1.7319E-6
	30.6354687928	1.2648E-5
This Algorithm	114.385515924	2.7249E-7
	30.6356739586	5.9507E-6

Table 3. Average Coordinates Comparison

In order to evaluate the reliability, the implementation of the algorithm was put forward in Windows XP platform, using the Visual C++ 6.0 as the Integrated Developing Environment. Besides, MapInfo data was adopted to show the vector map with the precision of 1:10000, from which the 3rd trunk road of Qingshan District in Wuhan was selected for experiment. Based on the requirement of the application, threshold value was set to 2 meters for corresponding tolerance band. An improvement on the result shown below could be made by based on the data provided in Table 2 and Table 3.

As shown in Table 2, the enhanced vector data compressing algorithm introduced in this paper had the least difference (8.5353E-4 in Table 2) in the comparison of the length value after the compression.

Similar statements could be made about Table 3, where the corresponding data of average coordinates had the least difference (2.7249E-7 difference in longitude and 5.9507E-6 difference in latitude in Table 3) after the compression by the enhanced vector data compressing algorithm.

Furthermore, the enhanced vector data compressing algorithm could process during the digitalization of the map with linear temporal complexity and linear spatial complexity in most of the time [17]. Thus, it had a fast compression speed.

6. Conclusion

This work addresses the design and realization of PPS. In PPS, a integrating positioning solution is proposed to overcome the problems both in standalone modes and in traditional integrating measurements. This method, which integrates the data both from GPS and digital compass, can provide continuous and reliable positioning, even under the circumstances of urban canyons, tunnels, and other GPS-denied environments. In order to improve runtime efficiency of PPS, we conduct a testing at the monitoring centor where the performance of native table file and memory table in MapX are compared using some commonly used operations. In addition, a new digital map compression algorithm is presented to provide low cost storage and rapid data transmission in PPS. This algorithm overcomes the problems that arise from the three traditional compressing algorithms, and can provide reliable and efficient solution.

There are a few improvements can be made in future research. First, despite the use of a combination of switching power supplies and linear power can improve the efficiency of power usage, it is still necessary to further improve the power component design. Second, the proposed PPS can only be positioned in the horizontal plane. It could be further investigated with tilt compensation based 3D positioning scheme.

Acknowledgments

The authors are thankful to Professor Kangling Fang from the College of Information Science and Engineering at Wuhan University of Science and Technology for her support and useful discussion. This work was supported in part by the Young Scientists Foundation of Wuhan University of Science and Technology (2012xz013), the Project (2008TD04) from Science Foundation of Wuhan University of Science and Technology, the Program of Wuhan Subject Chief Scientist (201150530152), the Educational Commission of Hubei Province (Q20101101, Q20101110), the projects (2009CDA034, 2009CDA136, 2008CDB345) from the Natural Science Foundation of Hubei Provincial, the project from Hubei Provincial Natural Science Funds for Distinguished Young Scholar of China (No. 2010CDA090), the project from Wuhan Chen Guang Project (No. 201150431095), and the Natural Science Foundation of China (60803160, 60975031, 61100055).

Author details

Xin Xu, Kai Zhang, Haidong Fu, Shunxin Li, Yimin Qiu and Xiaofeng Wang
School of Computer Science and Technology, Wuhan University of Science and Technology, China

7. References

[1] Altera, C. [2005a]. *Embedded Processor Solutions Overview*, http://www.altera.com.cn/technology/embedded/emb-index.html.

[2] Altera, C. [2005b]. *NIOS II Processor Reference handbook*, http://www.altera.com.cn/literature/litnios2.jsp.

[3] Bahr, A. [2009]. *Cooperative localization for autonomous underwater vehicles*, PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA.

[4] Bekir, E. [2007]. Introduction to modern navigation systems, *Technical report*, World Scientific Publishing Company.

[5] Boucheham, B. & Ferdi, Y. [2006]. Recursive Versus Sequential Multiple Error Measures Reduction: A Curve Simplification Approach to ECG Data Compression, *Computer Methods and Programs in Biomedicine* 81: 162–173.

[6] Caruso, M. [1997]. Applications of magnetoresistive sensors in navigation systems, *Technical report*, SAE Technical Paper 970602.

[7] D. Gaylor, G. L. & Key, K. [2005]. Effects of multipath and signal blockage on gps navigation in the vicinity of the international space station (iss), *Journal of The Institute of Navigation* 52: 61–70.

[8] Douglas, D. H. & Peucker, T. K. [1973]. Algorithm for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature, *The Canadian Cartographer* 10(2): 116–117.

[9] Dubey, R. [2008]. *Introduction to Embedded System Design Using Field Programmable Gate Arrays*, Springer, Berlin, Germany.

[10] Ebisch, K. [2002]. A Correction to the Douglas - Peucker Line Generalization Algorithm, *Computers and Geosciences* 28: 995–997.

[11] El-Rabbany, A. [2002]. *Introduction to GPS: The Global Positioning System*, Artech House Publishers.

[12] Farrell, J. [2008]. *Aided Navigation: GPS with High Rate Sensors*, McGraw-Hill Professional.

[13] Fu, W. [1997]. Expression of Knowledge and Organization of Knowledge base in Geographical Expert System, *Journal of Applied Science* 15(4): 482–489.

[14] Hu, P., Huang, X. & Hua, Y. [2002]. *Tutorial of Geographical Information System*, Wuhan University Press, Wuhan.

[15] Kehrer, D. & Bachu, D. [2011]. New generation high linearity navigation front-end devices covering gps and glonass, *Microwave Journal* 54: 86–96.

[16] Lapsley, P. [1997]. *DSP Processor Fundamentals: Architectures and Features*, IEEE Press, New York, USA.

[17] Liu, X. & Li, S. [2005]. Study on Subsection Douglas Algorithm with the Goniometry in Generalization, *Journal of Surveying and Mapping* 28(2): 51–52.

[18] M. Valtonen, L. Kaila, J. & Vanhala, J. [2011]. Unobtrusive human height and posture recognition with a capacitive sensor, *Journal of Ambient Intelligence and Smart Environments* 3: 305–332.

[19] MapInfo, C. [2004]. MapX online help.

[20] Nassar, S. [2003]. *Improving the inertial navigation system (INS) error model for INS and INS/DGPS applications*, PhD thesis, The University of Calgary.

[21] Q. Chang, D. Y. [2005]. *Vehicles Navigation Positioning Methods and Applications*, China Machine Press.

[22] Stringham, G. [2009]. *Hardware/Firmware Interface Design: Best Practices for Improving Embedded Systems Development*, Newnes: Waltham, MA, USA.

[23] U. Iqbal, A.F. Okou, A. N. [2008]. An integrated reduced inertial sensor system-riss/gps for land vehicle, *In Proceedings of the Position, Location and Navigation Symposium*, pp. 1014–1021.

[24] W. F. Abdelfatah, J. Georgy, I. U. N. A. [2012]. FPGA-based real-time embedded system for RISS/GPS integrated navigation, *Sensors* 12: 115–147.

[25] Wang, J. & Jiang, G. [2003]. Researching and Realization of the Quick Compression Method aimed at the Non-Topology Vector Data, *Journal of Surveying and Mapping* 32(2): 173–177.

[26] Wu, L. [2002]. *Geographical Information System: Principles and Applications*, Science Press, Beijing.

[27] Wu, L. & Shi, W. [2003]. *Theory and Algorithm of Geographical Information System*, Science Press, Beijing.

[28] Wu, Q. [1998]. Survey of gps/ins integrated navigation systems, *Navigation* 8: 1–10.

[29] Z. Lichuan, L. Mingyong, X. D. Y. W. [2009]. Cooperative localization for underwater vehicles, *4th IEEE Conference on Industrial Electronics and Applications*, pp. 2524–2527.

[30] Zheng, H. [1997]. *Theory of Mapping and Cartography Based Computer*, Institute of Surveying and Mapping, Zhengzhou.

[31] Zhong, S. & Gao, Q. [2004]. An Efficient Lossless Compression Algorithm for a kind of Two-Dimension Vector Maps, *Journal of System Simulation* 16(10): 2189–2194.

[32] Zhu, Z. [2006]. Geographical Information System Technology.
URL: *http://www.geog.ntu.edu.tw/course/gistech/*