A Simple Sequential Method for Integer Ambiguity Resolution in Real-Time GNSS Positioning

Shaocheng Zhang[1, 2], Samsung Lim[1] and Chris Rizos[1]

[1]*School of Surveying and Spatial Information Systems, University of New South Wales*
[2]*School of Geodesy and Geomatics, Wuhan University*

Shaocheng.zhang@student.unsw.edu.au

BIOGRAPHY

Shaocheng Zhang is a joint-training Ph.D. candidate at Wuhan University, P.R. China, and the University of New South Wales (UNSW), Australia. He received his M.Sc. degree in geodesy and geomatics from Wuhan University in 2006. His current research interests are mainly on network-RTK algorithms.

Samsung Lim is a Senior Lecturer in the School of Surveying and Spatial Information Systems at UNSW. For the past fifteen years his research has focused on the areas of GNSS and GIS. Samsung's research interests are in theoretical problems related to RTK-GPS and applying geo-spatial information technologies to real-world problems. Samsung received his B.A. and M.A. in Mathematics from Seoul National University, S. Korea, and his Ph.D. in Aerospace Engineering and Engineering Mechanics from the University of Texas at Austin, USA.

Chris Rizos is a graduate of UNSW; obtaining a Ph.D. in Satellite Geodesy in 1980. Chris is currently the Head of the School of Surveying & Spatial Information Systems at UNSW. Chris has been researching the technology and applications of GPS since 1985, and established over a decade ago the Satellite Navigation and Positioning group at UNSW, today the largest and best known academic GPS and wireless location technology R&D laboratory in Australia. Chris is the Vice President of the International Association of Geodesy (IAG), and a member of the Governing Board of the International GNSS Service. Chris is a Fellow of the IAG and of the Australian Institute of Navigation.

ABSTRACT

The successful resolution of integer ambiguities in Global Navigation Satellite System (GNSS) carrier-phase measurements is an essential but challenging task, especially where real-time kinematic (RTK) positioning is concerned. In this paper, a simple sequential least squares method for integer ambiguity resolution is presented in order to demonstrate how an open source toolkit - known as GPSTk - can be used by researchers as a reusable resource for developing sophisticated GNSS data processing software. GPSTk provides a set of GPS processing algorithms for static baseline computation, standard point positioning, atmospheric models, time and coordinate transformations, I/O routines, and many more. This paper describes a modification of the original solution for static GPS baselines to enable RTK positioning. A simple sequential least squares algorithm is devised and implemented to fix the integer ambiguities in kinematic mode. The LAMBDA method also employed for rapid ambiguity resolution, hence its take about 30 seconds to fix the ambiguities, and after ambiguity initialisation the resultant RTK positioning accuracy is comparable to that obtained using commercial software. The sequential method can be used not only for rover receiver initialisation, but also for network-RTK ambiguity resolution.

INTRODUCTION

GPSTk - the GPS Toolkit (cf. The University of Texas at Austin, 2004; Tolman and Harris, 2004) - is a package of GPS data processing routines written in the C++ language designed to facilitate scientific studies and computations involving the Global Positioning System (GPS). It is a library that can be linked to software developers' applications to allow them to use some or all of these routines for GPS data processing, using RINEX data files, time conversions, ephemeris calculations, atmospheric delay modelling for ionosphere and troposphere, position determination, matrix/vector algebra, and much more. It has been released under the Lesser General Public License (Free Software Foundation, Inc., 1999), allowing also commercial software development to benefit from it. Anyone interested in GPS software development can download the source code and participate in discussions with other GPS software developers on GPSTk's official website. In addition, researchers could make their own contribution to this open source package, by uploading their own software to the open source platform and sharing their work with others.

A GPS-RTK program - *kvecsol*, has been developed based on GPSTk, using a simple sequential method to improve computer efficiency and speed up the LAMBDA method for ambiguity resolution.

Ordinarily it would take a considerable amount of time to develop sophisticated GPS-RTK software, from the basic observation file input, coordinate transformation, calculation of satellite position, generating the matrices for computing parameters, and so on. Fortunately the GPSTk package has all of the above functions within its library, hence freeing the software developer from having to code the routines for these low level tasks. As the GPSTk code is well documented and easy to follow, it takes only a few weeks to develop a sophisticated program for GPS-RTK positioning.

The main purpose for this paper is to introduce this package so as to aid GPS researchers by offering a more efficient programming option. In addition the paper describes how the LAMBDA method of rapid ambiguity resolution has been implemented to support the first author's further research on network-RTK.

As a demonstration of the success of this coding task, two baselines with lengths 4.5km and 21km were tested. The test results demonstrate that after several seconds of ambiguity initialisation time, the developed software gives comparable accuracy results to commercial software.

GPSTK AND KVECSOL

Nowadays, there is considerable free software for GPS baseline computations. However, among the open source software most are developed with Matlab or FORTRAN. These two languages are not ideal from the point of view of source code readability and understandability. Furthermore, most of the open source software can only solve for static baselines.

The first author is a Ph.D candidate researching Network-RTK algorithms. In order to test the quality of the generated VRS data, an open source single baseline RTK program will be useful for simulating the RTK environment so as to debug the software and test the ambiguity initialisation time.

Too avoid spend too much time on low level programming - reading RINEX files, synchronisation of observations, matrix computations and so on - the first author has chosen an easy-to-understanding open source package to speed up the develop time of a simple RTK program. Thus, GPSTk is employed as the basic library for *kvecsol*. GPStk has three main modules as following[GPStk Twiki Site, 2008]:

- Core library. Provides the most robust, broadly useful, and platform independent code in the GPSTk. It provides a number of models and algorithms found in GPS textbooks and classic papers, such as solving for the user position or estimating atmospheric refraction. Common formats are supported as well, such as RINEX or SP3. There are several categories of functions that provide the base functionality for the GPSTk applications and for a number of other independent projects. The core library source code contains no dependencies outside of the GPSTk Core Library and Standard C++ and will build cleanly on all supported platforms.

- Auxiliary Libraries. These libraries contain code that could be useful in GPS processing but does not fit the description and portability of the core library. This code could contain highly specialised algorithms or be related to the message format of a specific receiver. The code could require libraries or system functions that are broadly available but not part of the C++ standard. Currently there are four auxiliary libraries for hardware interfacing or special requirements: *rxio* for receiver input and output library, *vplot* for vector graphics library, *geomatics* for precise position library and *procframe* for initial work at refactoring structure and prototype phase wind up support.

- GPSTk Applications. The libraries are the foundation for applications within the GPSTk suite. The applications provide greater functionality to support research and development. The applications are almost entirely console based (i.e., without a graphical user interface). There are a many successful and useful applications available, such as pseudo-range point positioning, residual analysis, ionospheric modelling, signal tracking simulation, data editing, and autonomous and relative positioning for navigation and surveying applications.

However, the GPStk's applications are mainly focused on post processing applications. The first author would like to promote GPSTk as an easy-to-understand, object oriented, open source package for researchers. That is why *kvecsol* as a GPS-RTK program was developed. At the present time it is not a real-time program built inside a GPS receiver, or a program reading data from a serial port and processing observations in real-time. It just reads observation files from both the reference station and rover station, synchronising epoch-by-epoch, and hence simulating a real-time environment.

MATHEMATICAL MODEL

A typical least squares solution procedure could be described as follows. Define the observation equation:

$$L = BX + \varepsilon \tag{1}$$

where L is a vector of GNSS observables, B is a linearised coefficient matrix for the unknown coordinate vector X , and ε is the stochastic error term. An approximate vector (or apriori information) X_0 can be used to determine the unknown variables. Let x and l be the differential terms:

$$\hat{x} = \hat{X} - X_0 , \tag{2}$$

$$l = L - BX_0 . \tag{3}$$

Applying Equations (2) and (3) to Equation (1), one can obtain a variance V from an estimated solution $\hat{x}$:

$$V = B\hat{x} - l \tag{4}$$

In order to obtain the minimum variance solution (or the least squares solution) with respect to a weight matrix P i.e. $V^T PV = \min$, $\hat{x}$ should satisfy the following equation:

$$\hat{x} = (B^T PB)^{-1} B^T Pl \tag{5}$$

In GPS positioning, pseudo-range and carrier-phase measurements for each satellite can be modelled as:

$$L_i^j = \rho_i^j + c \cdot \delta_i - c \cdot \delta^j - I_i^j + T_i^j + \lambda \cdot N_i^j + \varepsilon_\phi \tag{6}$$

$$P_i^j = \rho_i^j + c \cdot \delta_i - c \cdot \delta^j + I_i^j + T_i^j + \varepsilon_P \tag{7}$$

For double-differenced (DD) data, most of the error sources can be eliminated (or mitigated) and the observation equations will be:

$$\nabla\Delta L_i^j = \nabla\Delta\rho_i^j - \nabla\Delta I_i^j + \nabla\Delta T_i^j + \lambda \cdot \nabla\Delta N_i^j + \varepsilon_{\nabla\Delta\phi} \tag{8}$$

$$\nabla\Delta P_i^j = \nabla\Delta\rho_i^j + \nabla\Delta I_i^j + \nabla\Delta T_i^j + \varepsilon_{\nabla\Delta P} \tag{9}$$

where $\nabla\Delta I$ and $\nabla\Delta T$ are the DD residuals of the ionospheric and tropospheric delays, respectively.

The carrier-phase measurement can be written as:

$$L = \begin{bmatrix} b_{pos} & b_{ion} & b_{trop} & b_{amb} \end{bmatrix} \begin{bmatrix} x \\ I \\ T \\ n \end{bmatrix} + \varepsilon_\phi \tag{10}$$

When the baseline length is beyond several tens of kilometres in length, or the atmospheric conditions are unstable, this standard mathematical model may not be able to adequately account for the atmospheric delay. In such a case, the residuals may grow to more than half cycle of the wavelength, and false integer ambiguities may be computed.

On the other hand, in the case of a short baseline solution, the ionospheric and tropospheric delays could be considered to have been eliminated from the DD observation equations. The remaining unknown parameters will be only x and n . Then the observation matrix will be described as:

$$L = \begin{bmatrix} b_{pos} & b_{amb} \end{bmatrix} \begin{bmatrix} x \\ n \end{bmatrix} + \varepsilon_\phi \tag{11}$$

Another situation is ambiguity resolution for continuously operating reference station (CORS) permanent 'reference' receivers. As the position of the reference stations are assumed precisely defined, they are not unknown parameters. However, the distance between reference stations may be quite long - typically more than 20kms - so the unknown parameters will be I , T and n . As both the ionospheric and tropospheric delays are a function of their values at the zenith delay and the elevation angle (ionospheric delay also related to azimuth angle), one could combine these two elements. Then the observation matrix will be described by:

$$L = \begin{bmatrix} b_{atm} & b_{amb} \end{bmatrix} \begin{bmatrix} d_{atm} \\ n \end{bmatrix} + \varepsilon_{\phi} \tag{12}$$

For the case of Equations (11) and (12), if the aim is to generate high accuracy positions or atmospheric parameters – usually in order to generate network 'corrections' – fixing the ambiguities to their correct integer values is a crucial task.

In *kvecsol*, static observations are accumulated for initialisation at the beginning, then after the ambiguities have been fixed, observations read from every epoch will be considered as being independent and the position solved for each epoch.

In *kvecsol*, the initial value of x is obtained from the header of the RINEX file (although it could also be computed from the first epoch's pseudo-range observations), and the ambiguities are initialised with rounded values:

$$N_0 = round[(L - P)/\lambda] \tag{13}$$

The coefficient matrix B is defined as:

$$B = \begin{bmatrix} A & \lambda E \end{bmatrix} \tag{14}$$

E is an identity matrix and A is the coefficient matrix for position:

$$A = \begin{bmatrix} \alpha_1 & \beta_1 & \gamma_1 \\ \alpha_2 & \beta_2 & \gamma_2 \\ \cdots & \cdots & \cdots \\ \alpha_m & \beta_m & \gamma_m \end{bmatrix} \tag{15}$$

where

$$\alpha_i = \frac{X_0 - X^i}{\rho_0^i} - \frac{X_0 - X^{ref}}{\rho_0^{ref}} \tag{16}$$

$$\beta_i = \frac{Y_0 - Y^i}{\rho_0^i} - \frac{Y_0 - Y^{ref}}{\rho_0^{ref}} \tag{17}$$

$$\gamma_i = \frac{Z_0 - Z^i}{\rho_0^i} - \frac{Z_0 - Z^{ref}}{\rho_0^{ref}} \tag{18}$$

The weight matrix P is defined according to the elevation angle for each double-difference satellite pair, and the covariance is considered for all the DDs with respect to the same reference satellite observation:

$$P^{-1} = \begin{bmatrix} q_1 + q_{ref} & q_{ref} & \cdots & q_{ref} \\ q_{ref} & q_2 + q_{ref} & \cdots & q_{ref} \\ \vdots & \vdots & \ddots & \vdots \\ q_{ref} & q_{ref} & \cdots & q_m + q_{ref} \end{bmatrix} \tag{19}$$

where

$$q = \frac{1}{\sin(elev)} \tag{20}$$

AMBIGUITY RESOLUTION

In GPS relative position, observation equations can either be described as Equations (11) or (12). In both of these cases the unknown vector X consists of two parts. In the baseline case, the focus is on the position result x, while in the CORS baseline case, interest is in the atmospheric residual D,. However, in order to obtain high accuracy X or D, the ambiguity part N should be correctly resolved, for the integer property of ambiguities means that just one cycle error could result in several centimetres error in position x or atmospheric delay D. So the main task for high accuracy positioning should be to first fix the N vector to the correct integer values.

There are many ways to fix ambiguities. Hatch (2002) has classified them into three categories:

-Long duration static observation: This method is used in static mode where x is not changing. The reason that this method works is that long period observation sessions can average the multipath error and GPS receiver noise, and that due to B varying slowing (caused by GPS satellite motion) approximately 20 minutes are generally sufficient for B to change enough for the set of equations to yield observability. The long convergence time is the major drawback of this approach.

-GPS antenna special moving: GPS antenna swapping is described in Hatch (2002). Swapping the locations of two antennas causes the observability properties of the B matrix to change rapidly. However, it is rarely possible in real-time kinematic positioning or long baseline scenarios to implement this method.

-Searching methods: These methods require a few assumptions and have attracted the attention of many researchers. Numerous methods have been reported in the literature: Ambiguity Function Method (AFM), Fast Ambiguity Resolution Approach (FARA), Least Squares Ambiguity Search Technique, Cholesky Decomposition, Fast Ambiguity Search Filter (FASF), Least Square AMBiguity Decorrelation Adjustment (LAMBDA), and Integrated Ambiguity Resolution Method.

In this paper the authors have adopted the LAMBDA method for rapid ambiguity resolution. Using the LAMBDA method, the integer least squares ambiguity estimates are computed in two steps. First the ambiguities are de-correlated by means of the Z-transformation. Then the integer minimisation problem is solved by a discrete search over an ellipsoidal region, the so-called ambiguity search ellipsoid. Both Equation (8) and (9) can be described by the following equation:

$$L = a \cdot x + b \cdot n + e \tag{21}$$

where L are the double-differenced observations, x and n are the unknown parameters, e is the observation noise, and a and b are the corresponding coefficients. The DD carrier-phase ambiguities are expressed in units of cycles rather than range. They are known to be integers. x therefore consist of the remaining unknown parameters, such as baseline components (coordinates) and possibly DD atmospheric delay parameters (troposphere, ionosphere).

The procedure can be divided into three steps:

-Firstly, solve the observation matrix and obtain the float values of the unknown parameters (both $\hat{x}$ and $\hat{n}$), together with their variance-covariance matrices $Q_{\hat{x}}$ and $Q_{\hat{n}}$,

-The second step is to validate the ambiguity parameters $\hat{n}$ to integer values, and make sure the integer ambiguities are unique and correct. In this step a de-correlation transformation for $Q_{\hat{n}}$ has been performed, and the search space for the integer ambiguities has been reduced to as a small an area as possible. This makes the ambiguity fixing more efficient (and rapid).

-Finally, use the fixed ambiguities to correct the float parameters $\hat{x}$ and the corresponding variance-covariance matrix:

$$\breve{x} = \hat{x} - Q_{\hat{x}\hat{n}} Q_{\hat{n}}^{-1} (\hat{n} - \breve{n}) \tag{22}$$

$$Q_{\breve{x}} = Q_{\hat{x}} - Q_{\hat{x}\hat{n}} Q_{\hat{n}}^{-1} Q_{\hat{n}\hat{x}} \tag{23}$$

More information about the LAMBDA method can be found in Teunissen (1993). The authors implemented the method using the GPSTk data structure of Matrix and Vector using the C++ language.

SEQUENTIAL LEAST SQUARES METHOD

For GPS observations, multiple epochs of observations are needed to fix the ambiguities to the correct integer values, if using the standard least squares method. The B matrix will increase as the processed measurements increase, and transposing and inverting high dimensional matrices will be a significant computational burden. Sequential least squares could improve the computational and storage efficiency.

Take two sequential static epochs, t_1 and t_2 with the same satellite constellation, which means the unknown parameters x_{t_1} and x_{t_2} are the same for these two epochs. The observation matrix can be written as:

$$\begin{aligned} V_{t_1} &= B_{t_1}\hat{x} - l_{t_1} \\ V_{t_2} &= B_{t_2}\hat{x} - l_{t_2} \end{aligned} \tag{24}$$

In the standard least squares method，one obtains:

$$V = \begin{bmatrix} V_{t_1} \\ V_{t_2} \end{bmatrix},\ B = \begin{bmatrix} B_{t_1} \\ B_{t_2} \end{bmatrix},\ P = \begin{bmatrix} P_{t_1} \\ P_{t_2} \end{bmatrix} \text{ and } l = \begin{bmatrix} l_{t_1} \\ l_{t_2} \end{bmatrix}$$

The solution will be:

$$\hat{x} = N^{-1}W \tag{25}$$

where

$$N = B^T PB \tag{26}$$

$$W = B^T Pl \tag{27}$$

If there are **n** common visible satellites, there will be **n-1** DD observation equations for that epoch.

With the standard least squares method if there are **m** epochs in the initialisation session, there will be **(n-1)×m** DD observation equations, the B matrix will be a **(n-1)×m** rows matrix, and the N matrix will be a **(n-1)×m** by **(n-1)×m** matrix. While as **m** gets larger the inverse computation for the N matrix will consume ever more computer resources.

Although:

$$N \neq N_1 + N_2 \tag{28}$$

$$W \neq W_1 + W_2 \tag{29}$$

they produce the same solution vector x, which means that the multiplication results for N^{-1} and W are the same.

So by using the sequential method one obtains:

$$N^{-1}W = (N_{t_1} + N_{t_2})^{-1}(W_{t_1} + W_{t_2}) \tag{30}$$

As a sequential method, the multi-epoch data is processed in a much more efficient way as follows:

$$N = \sum_{i=1}^{n} N_i \tag{31}$$

$$W = \sum_{i=1}^{n} W_i \tag{32}$$

$$\hat{x} = N^{-1}W \tag{33}$$

In this way, it takes only two **m×m** matrices to store N and N_i, and two **m×1** matrices for W and W_i storage. Hence there is only one **m×m** matrix for the inverse computation in the whole algorithm.

Note that during the initialisation time there are several situations that may cause x to change, and then the sequential method will be disrupted:

(1) Reference satellite changes. In the RTK model, the reference satellite is commonly chosen according to the elevation angle, normally the highest satellite, but as the satellite is moving, the reference satellite may not always be the highest satellite during the initialisation period. This problem can be easily solved by forcing the reference satellite not to change during the initialisation period.

(2) Satellite number changes. During the initialisation period, new satellites rise up and some low elevation satellites may go down, causing the unknown parameter x to change. In *kvecsol*, a 37×37 static array (32 for ambiguity, 3 for position and 2 for troposphere zenith delay) is used to store the N elements for accumulation and the initial ambiguity for every satellite is initialised at the epoch it appears for the first time (not necessarily the first epoch).

(3) Satellite lost and re-tracked. Sometimes during the initialisation period tracking on a satellite may be lost for several seconds, and then reacquired. For this situation, when the satellite is re-tracked, the first thing is to check for cycle slips. If there are no cycle slips, the satellite's element of the N matrix continues accumulating, otherwise the satellite's element in the N matrix will need to be re-initialised.

(4) The last situation is that sometimes lock on the reference satellite is lost, and then there is no choice but to re-initialise all the satellites again. Switching reference satellite may be another option, but if correlations are considered in the weight matrix, it will be too complicated to implement.

TEST RESULTS

The algorithm described above has been implemented as *kvecsol* – a program based on GPSTk. It has been tested with data from several baselines. The first baseline is from the CORS receivers UNSW to CHIP with a length of about 4.5 kilometres. The position component values for north, east and height are shown in Figures 1, 2, 3.

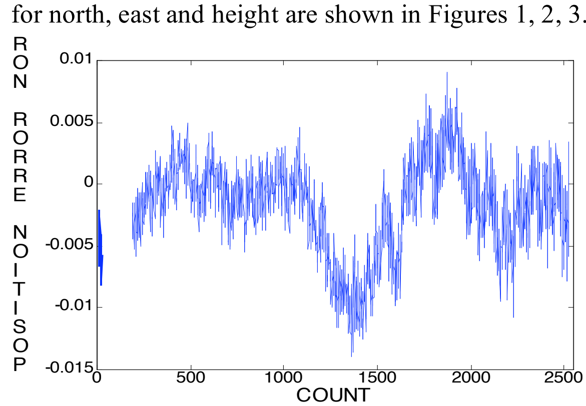

Figure 1 North component of baseline (UNSW-CHIP)

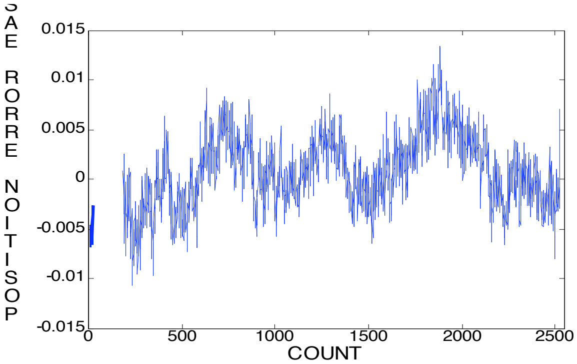

Figure 2 East component of baseline (UNSW-CHIP)

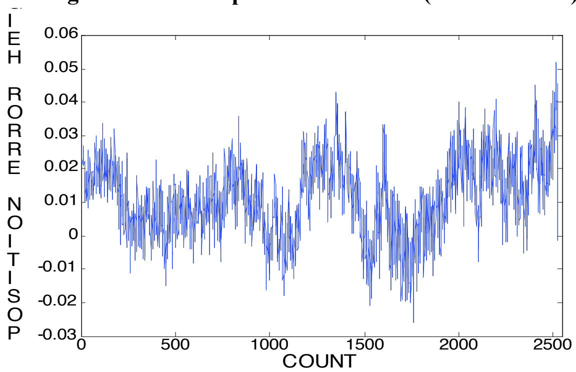

Figure 3 Height component of baseline (UNSW-CHIP)

The second baseline is from CORS receivers CHIP to VLWS with length of about 21 kilometres. The position components for north, east and height are shown in Figures 4, 5, 6.

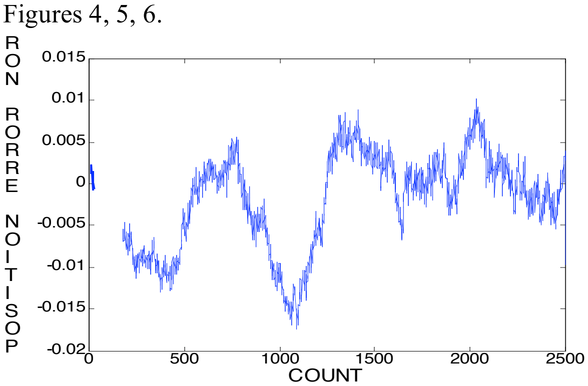

Figure 4 North component of baseline (CHIP-VLWD)

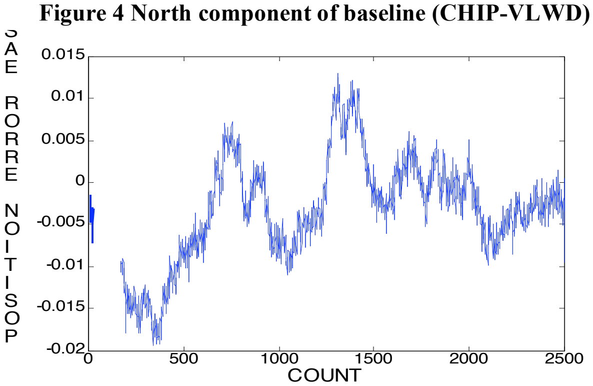

Figure 5 East component of baseline (CHIP-VLWD)

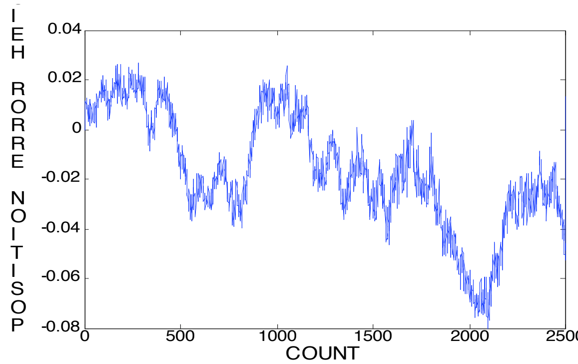

Figure 6 Height component of baseline (CHIP-VLWD)

More statistical information regarding the position accuracy for these baselines is given in Table 1.

Table 1. Position results with *kvecsol*

Baseline Length (m)	Initial Time (s)	Duration (s)	Std Deviation (cm)		
			N	E	U
4436.9	28	2519	0.42	0.37	1.61
20738.7	50	2497	0.59	0.71	2.83

The same datasets are processed with the commercial software Leica Geo Office (LGO) in kinematic mode, and the results are listed in Table 2.

Table 2. Position results with Leica Geo Office

Baseline Length (m)	Initial Time (s)	Duration (s)	Std Deviation (cm)		
			N	E	U
4436.9	47	2519	0.32	0.27	0.88
20738.7	106	2497	0.53	0.54	1.52

It shows that LGO took more initialisation time than *kvecsol* and with better accuracy. That may be because LGO uses more critical criteria to validate the ambiguities, and the LGO defines the weight matrix more rigorously.

SUMMARY

Firstly, the open source package GPSTk has been tested, although the auxiliary libraries and applications may crash in some cases of platform dependence, the core library package has proven to be useful and reliable. A simple sequential least squares algorithm and the LAMBDA method were implemented base on GPSTk's core library, and coordinates of comparable accuracy to those obtained from commercial GPS data processing software were obtained. Currently, *kvecsol* can only fix the ambiguities during a static initialisation period, however further work will extend the sequential method to operate in on-the-fly ambiguity fixing mode.

REFERENCES

[1] Free Software Foundation, Inc. (1999). The GNU Lesser Public License. http://www.gnu.org/copyleft/lesser.txt. Version 2.1, viewed March 1, 2006.

[2] GPStk TWiki Site (2008). http://www.gpstk.org. (accessed 10 September 2008).

[3] B. Tolman, R.B. Harris, T. Gaussiran, D. Munton, J. Little, R. Mach, S. Nelsen, B. Renfro, & D. Schlossberg (2004). "The GPS Toolkit -- Open Source GPS Software". 17th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2004), Long Beach, California, 21-24 September, 2044-2053.

[4] M. Vermeer, & M. Vaisanen (2006). "Geodetic Baseline GPS Processing by a Simple Sequential Technique". 19th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2006), Fort Worth, Texas, 26-29 September, 2877-2882.

[5] X.W. Chang, C.C. Paige, & L. Yin (2004). "Code and Carrier Phase Based Short Baseline GPS Positioning: Computational Aspects". GPS Solutions, 7(4), 230-240.

[6] Y. Yun-chun , R.R. Hatch, & R.T. Sharpe (2003). “Minimizing the Integer Ambiguity Search Space for RTK”. Wuhan University Journal of Natural Sciences, 8(2), 485-491.

[7] Y. Yun-Chun, R.T. Sharp, & R.R. Hatch (2002). “A Fast Ambiguity Resolution Technique for RTK Embedded Within a GPS Receiver”. 15th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS 2002), 24-27 September, 945-952.

[8] P.J.G. Teunissen (1993). “Least-Squares Estimation of the Integer GPS Ambiguities”. Invited lecture, Section IV Theory and Methodology, IAG General Meeting, Beijing, China. Also in Delft Geodetic Computing Centre LGR series, No. 6, 16pp.

[9] P.J. Jonge de, & C.C.J.M. Tiberius (1996). “The LAMBDA Method for Integer Ambiguity Estimation: Implementation Aspects”. *Delft Geodetic Computing Centre LGR series,* No. 12.