The GPS Toolkit

Learn how GPS works and get a sharper fix on your position with this freely licensed library.

BY BRIAN W. TOLMAN AND BEN HARRIS

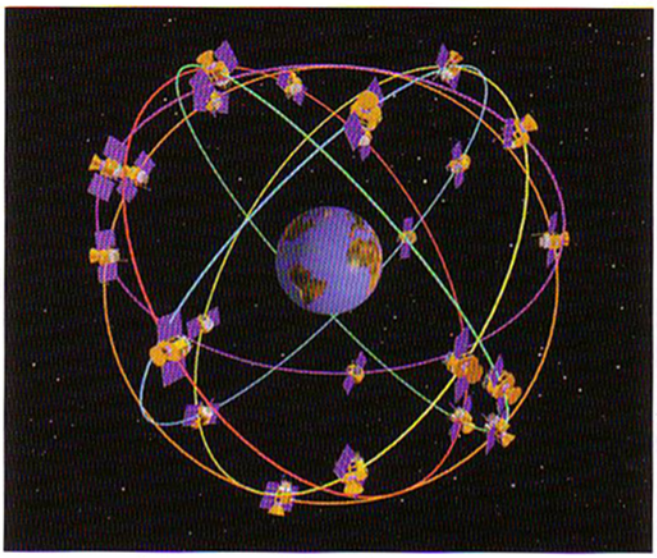

Figure 1. GPS Satellite Constellation Image from the Aerospace Corporation Web Site (www.aero.org/news/current/gps-orbit.html)

Explosive is perhaps the best term to describe the growth of the Global Positioning System (GPS) market in recent years. Contributing factors are numerous, and perhaps the most dramatic are economic: access to GPS is absolutely free and the cost of hardware continues to plummet. As a result, the GPS user can choose from a variety of devices that provide a position estimate. GPS has long been used, however, to explore topics beyond positioning; space weather, precise timing and continental drift are but three examples.

In order to use GPS for advanced topics or simply for improved positioning, the raw observations collected by the GPS receiver must be processed. In the past, the nuts and bolts of such processing have been left up to proprietary software. Now, a project called the GPS Toolkit, or GPSTk, is available under the LGPL to the Open Source and research communities. GPSTk is the by-product of GPS research conducted at the Applied Research Laboratories of the University of Texas at Austin (ARL:UT) since before the first satellite launched in 1978. It is the combined effort of many software engineers and scientists. Recently, the research staff at ARL:UT has decided to open source much of their basic GPS processing software as the GPSTk.

The Global Positioning System

The Global Positioning System actually is a US government satellite navigation system that provides a civilian signal. As of this writing, the signal is broadcast simultaneously by a constellation of 29 satellites, each with a 12-hour orbit. From any given position on the Earth, 8–12 satellites usually are visible at a time.

GPS in a Nutshell

Each satellite broadcasts spread spectrum signals at 1,575.42 and 1,227.6MHz, also known as L1 and L2, respectively. Currently, the civil signal is broadcast only on L1. The signal contains two components, a time code and a navigation message. By differencing the received time code with an internal time code, the receiver can determine the distance, or range, that the signal has traveled. This range observation is offset by errors in the (imperfect) receiver clock; therefore, it is called a pseudorange. The navigation message contains the satellite ephemeris, which is a numerical model of the satellite's orbit.

GPS receivers record, besides the pseudorange, a measurement called the carrier phase, or phase. The phase also is a range observation like the pseudorange is, except it has an unknown constant added to it, the phase ambiguity. It also is much smoother, having about 100 times less measurement noise than the pseudorange, which makes it useful for precise positioning. Because of the way it is measured, the phase is subject to random, sudden jumps. These discrete changes always come in multiples of the wavelength of the GPS signal and are called cycle slips.

The Position Solution

The standard solution for the user location requires a pseudorange measurement and an ephemeris for each satellite in view. At least four measurements are required, as there are four unknowns: three coordinates of position plus the receiver clock offset. The basic algorithm for the solution is described in the official GPS Interface Control Document, ICD-GPS-200. The position solution is corrupted due to two sources of error, errors in the observations and errors in the ephemeris.

Reducing Measurement Errors

The GPS signal travels through every layer of the Earth's atmosphere. Each layer affects the signal differently. The ionosphere, which is the high-altitude, electrically charged part of the atmosphere, introduces a delay, and therefore a range error, into the signal. The delay is frequency-dependent, so it can be computed directly if you have data on both the GPS frequencies. There also is a delay due to the troposphere, the lower part of the atmosphere. This delay also can be modeled and removed. Many other errors are associated with the GPS signal. Multipath reflections and relativistic effects are two examples.

More precise applications reduce the effect of error sources by a technique referred to as differential GPS (DGPS). By differencing measurements simultaneously collected by the user and a nearby reference receiver, the errors common to both receivers (most of them) are removed. The result of DGPS positioning is a position relative to the reference receiver;

adding the reference position to the DGPS solution results in the absolute user position.

The alternative to DGPS is to model and remove errors explicitly. Creating new and robust models of phenomena that affect the GPS signal is an area of active research at ARL:UT and other laboratories. The positioning algorithm can be used to explore such models. Essentially, the basic approach is to turn the positioning algorithm inside out to look at the corrections themselves. For example, observations from a network of receivers can create a global map or model of the ionosphere.

Improved Ephemerides

The GPS position solution can be improved by using a better satellite ephemeris. The US National Geospatial-Intelligence Agency (NGA) generates and makes publicly available a number of precise ephemerides, which are more accurate satellite orbits. Satellite orbits described by the broadcast navigation message have an error on the order of meters, and the precise ephemeris has decimeter accuracy. The International GPS Service (IGS) is a global civil cooperative effort that also provides free precise ephemeris products. Global networks of tracking stations produce the observations that make generation of the precise ephemerides possible.

GPS Data Sources

GPS observation data from many tracking stations are available freely on the Internet. Many such stations contribute their data to the IGS. In addition, many networks of stations also post their data to the Internet, such as the Australian Regional GPS Network (ARGN) and global cooperatives including NASA's Crust Dynamics Data Information System (CDIS).

GPS File Formats

Typically, GPS observations are recorded in a standardized format developed by and for researchers. Fundamental to this format is the idea that the data should be independent of the type of receiver that collected it. For this reason, the format is called receiver independent exchange, or RINEX. Another format associated with GPS is SP-3, which records the precise ephemeris. The GPSTk supports both RINEX and SP-3 formats.

GPS Receivers and Open Source

GPS receivers have become less expensive and more capable over the years, in particular handheld and mobile GPS receivers. The receivers have many features in common. All of the receivers output a position solution every few seconds. All receivers store a list of positions, called waypoints. Many can display maps that can be uploaded. Many can communicate with a PC or handheld to store information or provide position estimates to plotting software.

Typically, communication with a PC and other system follows a standard provided by the National Marine Electronics Association, called NMEA-0183. NMEA-0183 defines an ASCII-based format for communication of position solutions, waypoints and a variety of receiver diagnostics. Here is an example of a line of NMEA data, or sentence:

```
$GPGLL,5133.81,N,00042.25,W*75
```

GPS Drive also can be linked to the Festival application to get driving directions in the form of speech output.

The data here is a latitude, longitude fix at 51° 33.81 min North, 0° 42.25 min West. The last part is a checksum.

As a public standard, the NMEA-0183 format has given the user of GPS freedom of choice. NMEA-0183 is the format most typically used by open-source applications that use receiver-generated positions.

Closed standards also are common. SiRF is a proprietary protocol licensed to receiver manufacturers. Many receiver manufacturers implement their own binary protocols. Although some of these protocols have been opened to the public, some have been reverse engineered. GPSBabel is an open-source project to communicate with consumer-grade receivers. The Sharc Project is a similar project to provide communication with survey-grade receivers.

A number of interesting open-source applications are available that utilize consumer-grade receivers. With one, you can use open-source applications to navigate in your car. The GPS Drive Project helps you do that, using a graphical map. GPS Drive also can be linked to the Festival application to get driving directions in the form of speech output. Internet sites such as WiGLE.net have lists of the geographic coordinates of open wireless LANs; you can use your GPS unit to find these.

Traditionally, DGPS is accomplished with two or more receivers that communicate position information with radio waves. You can do DGPS over IP now, using open-source applications. The open-source project called gpsd essentially broadcasts NMEA-0183 sentences over TCP/IP. The gps3d Project, which allows you to visualize your position and the configuration of GPS in 3-D, also can utilize a gps3d server.

All of these applications are based on standard positioning. To move your positioning capability to the next level, you have to work directly with the observations made by the receiver. Only a few open-source or freely available programs exist that give the user this freedom. OpenSourceGPS is a project to create a GPS receiver based on the Zarlink chipset. teqc from UNAVCO performs quality assurance and processes raw data from receivers to generate RINEX, but it is closed source. In contrast, the purpose of the GPSTk is to give the user the ability to manipulate not only GPS observations but also to improve the processing algorithms.

The GPS Toolkit

The GPS Toolkit (GPSTk) is coded entirely in ANSI C++. It is platform-independent and has been built and tested on Linux, Solaris and Microsoft Windows. Everything needed to write standalone, console-based programs is included, along with several complete applications.

The design is highly object-oriented. Everything is contained in the namespace gpstk::. For example, reading and writing a RINEX observation file is as simple as this:

```
// open, read and re-write a RINEX file
using namespace gpstk;
// input file stream
RinexObsStream rin(inputfile);
// output file stream
RinexObsStream rout(outputfile,
ios::out|ios::trunc);
DayTime nextTime; //Date/time object
RinexObsHeader head; //RINEX header object
RinexObsData data; //RINEX data object

// read the RINEX header
rin >> head;
rout.header = rin.header;
rout << rout.header;

// loop over all data epochs
while (rin >> data) {
nextTime = data.time;
// change obs data&
rout << data;
}
```

The core capability of the library is built around RINEX file I/O. It also includes a complete date and time class to manipulate time tags in GPS and many other formats.

In addition to the RINEX I/O, GPSTk includes classes for handling geodetic coordinates (latitude and longitude) and GPS ephemeris computations. There also is a complete template-based Matrix and Vector package. And, of course, there are GPS positioning and navigation algorithms, including several tropospheric models.

Finally, several standalone programs are included in the distribution. Included are utilities to validate or modify RINEX files, a summary program, a utility to remove or modify observations, a phase discontinuity corrector and a program to compute standard errors and corrections, such as the total electron content (TEC) of the ionosphere along the signal path.

Getting Started with the GPS Toolkit

The GPS Toolkit is available for download as a tarball (see the on-line Resources section). To build the toolkit you need to use jam, a replacement for make, and Doxygen, a source code documentation generator. The entire build sequence looks like the following:

```
tar xvzf gpstk-1.0.tar.gz
cd gpstk
jam
doxygen
su
jam -sPREFIX=/usr install
```

This sequence builds and installs the GPSTk dynamic and shared libraries, as well as the header files, in the /usr tree. In addition, a doc subdirectory is created, containing HTML-based documentation of the GPSTk library.

Below are three example applications of the GPSTk created

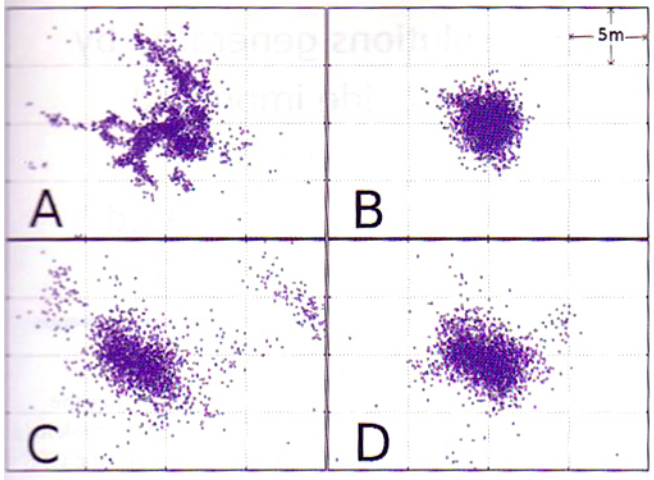

Figure 2. Left: positions from a GPS receiver. Right: positions generated using GPSTk algorithms.

at ARL:UT. The second example actually is distributed as an application with the GPSTk.

Enhanced Positioning

Position solutions generated by the GPSTk provide improved precision and robustness compared to those generated by a GPS receiver. Figure 2 illustrates the benefits; each axis extends from –10 to 10 meters.

Plot A shows position computations and how they vary along the East and North directions. Such results are representative of solutions created with a consumer-grade GPS receiver. Plot B shows how the position estimate improves when atmospheric delays are accounted for. Direct processing not only improves precision, but it also increases robustness. Plot C shows the effect of a faulty satellite. The faulty satellite is detected and removed using the GPSTk in Plot D.

Carrier-Phase Discontinuity Correction

An important problem in GPS data processing involves discontinuities in the carrier phase. Before phase data can be used, cycle slips must be found and fixed. The GPSTk distribution includes an application called a discontinuity corrector that

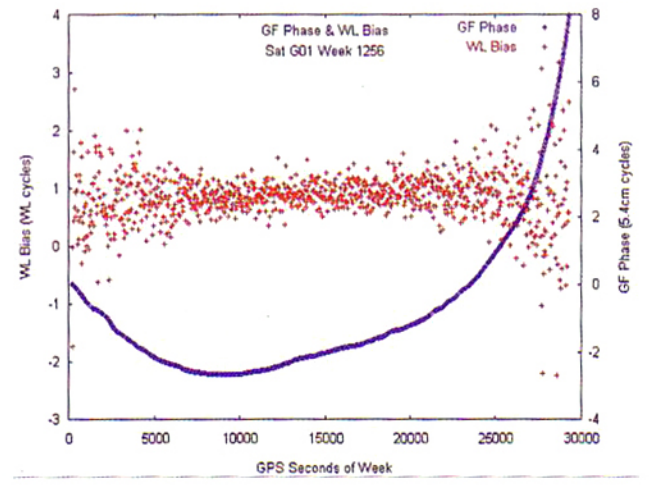

Figure 3. Normal wide-lane (red) and geometry-free (blue) phases for one satellite.

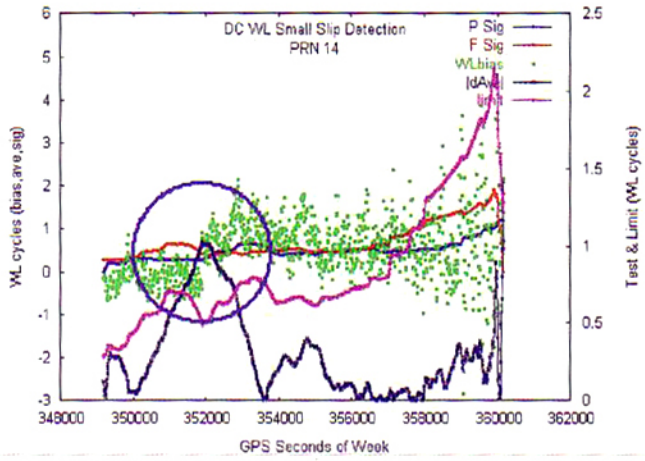

Figure 4. Slip detected (blue circle) in the wide-lane data (green) where test quantity (dark blue) is larger than limit (magenta).

does just that. This feature is available in the library as well.

The GPSTk discontinuity corrector works by forming two useful linear combinations of the dual-frequency phase data, called the wide-lane phase bias and the geometry-free phase. An example of these for normal data is shown in Figure 3. The wide-lane bias (red) is noisy but has a constant average. The geometry-free phase does not depend on the receiver-satellite geometry, but it depends strongly on the ionospheric delay. In

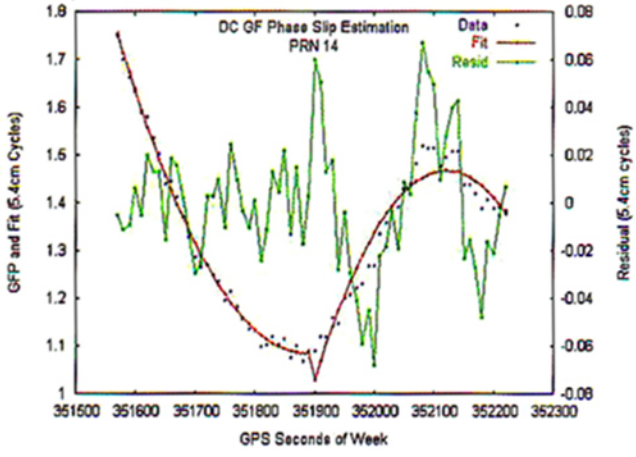

Figure 5. Estimation of the Cycle Slip Size

fact, it is proportional to that delay. Normally, the ionosphere is quiet and smooth, but at times it can be active and rough; then this quantity can vary wildly. The geometry-free phase and the wide-lane noise increase at both ends of the dataset, because the satellite is rising or setting there. Consequently, the signal must travel through more atmosphere.

The discontinuity corrector works by first looking for slips in the wide-lane phase bias; Figure 4 illustrates a case in which it found one. When a slip in the wide-lane slip is found, the code turns to the geometry-free phase and looks for the slip there. To estimate the size of the slip, low-order polynomials are fit to the data on each side of the slip, extrapolated to the point where the slip occurred and then differenced.

Satellite Position Interpolation

Another GPSTk application at ARL:UT involves a satellite in low-earth orbit that carries a GPS receiver. This satellite collects GPS data both for the satellites above it, referred to as top-side data, and those visible below it, or bottom-side data. The GPS signal for the bottom-side data has a long path length through the atmosphere, which is ideal for remote sensing of the atmosphere. The top-side data is used for computing the LEO satellite's rapidly changing position as it orbits Earth. A problem arises here as the top-side data is collected with a lower data rate (10 seconds) than on the bottom-side (1 second), yet the

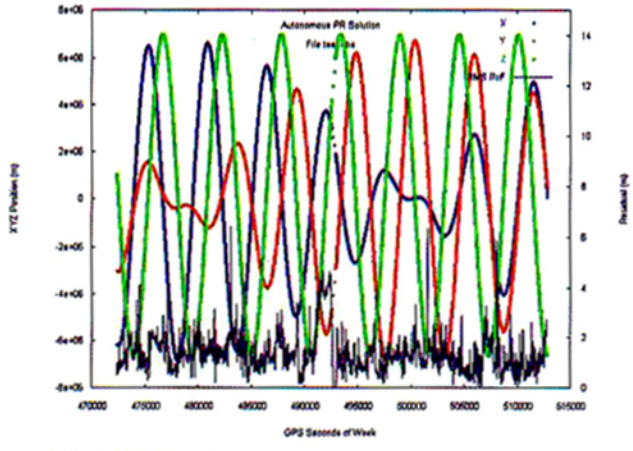

Figure 6. Position Interpolation

Position solutions generated by the GPSTk provide improved precision and robustness compared to those generated by a GPS receiver.

position of the LEO satellite is needed for processing the bottom-side data at the higher data rate. To solve the problem, a program was written, using only GPSTk, which reads the GPS data, computes the LEO position with the top-side data and then interpolates those positions to 1-second epochs. The result is shown in Figure 6, a plot of the position of the LEO satellite as it orbits Earth.

The Future of GPSTk

Open-source GPS processing, on the scale anticipated for the GPS Toolkit, is unprecedented; we are excited by the prospect of what could develop. GPSTk potentially has a broad range of audiences. Universities can use the GPSTk to process GPS data with open-source code. Embedded developers can develop software to perform GPS positioning and to read, write and edit RINEX data files. Researchers may find that this code is an excellent foundation for GPS receivers implemented entirely in software, called software receivers.

Although the growth of the GPSTk will depend strongly on user feedback and participation, changes also will be driven by shifts in the satellite navigation arena. In the near term, the first satellite to provide dual-frequency pseudoranges to civilians is scheduled for launch in 2005. Furthermore, the European community is creating Galileo, which will provide a public-regulated service compatible with GPS, essentially augmenting the current constellation with a new one. In the long term, GPS will have new signals in the L5 and M code. GPSTk, with its emphasis on fundamental observations, can provide the basis to explore and exploit these changes.

It is our hope that university students, laboratory researchers, system engineers and software developers will contribute to, as well as benefit from, the GPS Toolkit. We already have seen many benefits to using this code within our lab and believe that the GPSTk will inspire a number of innovative GPS applications.

Resources for this article: www.linuxjournal.com/article/7651.

Dr Brian W. Tolman is a research scientist at Applied Research Laboratories, The University of Texas, with 18 years experience in GPS-related research, data analysis and software development. He holds a PhD in theoretical physics from The University of Texas at Austin.

Ben Harris is an Engineering Scientist at UT Austin. When he is not researching GPS, studying for a PhD or spending time with his beautiful family, he reprograms animatronic fishes to perform scenes from *Pulp Fiction*.