The GPS Toolkit

A User's Guide for Scientists, Engineers and Students

Tracie Conn, Tom Gaussiran, R. Benjamin Harris, Jon Little
Richard Mach, David Munton, Brent Renfro, Brian Tolman

Applied Research Laboratories, The University of Texas at Austin

Martin Vermeer

Geophysics Department, Helsinki University of Technology

August 7, 2006
GPSTk Revision 74

Contents

I Theory 3

1 The Global Positioning System in a Nutshell 5
- 1.1 GPS in a Nutshell . . . 5
  - 1.1.1 The Position Solution . . . 5
- 1.2 GPS Data Sources . . . 6
  - 1.2.1 GPS File Formats . . . 7
  - 1.2.2 Receiver Protocols . . . 7

2 GPS File Formats 9
- 2.1 FIC . . . 9
- 2.2 RINEX . . . 9

3 Converting Coordinates & Time 11
- 3.1 Transformations . . . 11
- 3.2 Time Systems . . . 11
  - 3.2.1 Solar & Sidereal Time . . . 11
  - 3.2.2 Atomic Time . . . 12
  - 3.2.3 Time Formats . . . 12
  - 3.2.4 GPS Time . . . 13
  - 3.2.5 Z-Count . . . 14
- 3.3 Earth Fixed Coordinates . . . 14
  - 3.3.1 ECI to ECF . . . 14
  - 3.3.2 WGS-84 . . . 15
  - 3.3.3 Coordinate Systems . . . 15
- 3.4 References . . . 18

II Usage, Examples & Notes 19
- 3.5 *calgps* . . . 22
  - 3.5.1 Overview . . . 22
  - 3.5.2 Usage . . . 22
  - 3.5.3 Examples . . . 22
  - 3.5.4 Notes . . . 23
- 3.6 *ficafic ficfica* . . . 24

3.6.1 Overview . . . 24
3.6.2 Usage . . . 24
3.6.3 Examples . . . 24
3.6.4 Notes . . . 24
3.7 *fic2rin* . . . 25
3.7.1 Overview . . . 25
3.7.2 Usage . . . 25
3.7.3 Examples . . . 25
3.7.4 Notes . . . 25
3.8 *mdp2fic mdp2rinex* . . . 26
3.8.1 Overview . . . 26
3.8.2 Usage . . . 26
3.8.3 Examples . . . 26
3.8.4 Notes . . . 26
3.9 *mergeRinObs mergeRinNav mergeRinMet mergeFic* . . . 27
3.9.1 Overview . . . 27
3.9.2 Usage . . . 27
3.9.3 Examples . . . 27
3.9.4 Notes . . . 27
3.10 *navdmp* . . . 28
3.10.1 Overview . . . 28
3.10.2 Usage . . . 28
3.10.3 Examples . . . 28
3.10.4 Notes . . . 29
3.11 *poscvt* . . . 30
3.11.1 Overview . . . 30
3.11.2 Usage . . . 30
3.11.3 Examples . . . 30
3.11.4 Notes . . . 30
3.12 *PRSolve* . . . 31
3.12.1 Overview . . . 31
3.12.2 Usage . . . 31
3.12.3 Examples . . . 32
3.12.4 Notes . . . 32
3.13 *reszilla* . . . 33
3.13.1 Overview . . . 33
3.13.2 Usage . . . 33
3.13.3 Examples . . . 34
3.13.4 Notes . . . 34
3.14 *RinexDump* . . . 35
3.14.1 Overview . . . 35
3.14.2 Usage . . . 35
3.14.3 Examples . . . 35
3.14.4 Notes . . . 35
3.15 *rinexpvt* . . . 36
3.15.1 Overview . . . 36

3.15.2 Usage . . . 36
3.15.3 Examples . . . 36
3.15.4 Notes . . . 37
3.16 *rinexthin* . . . 38
3.16.1 Overview . . . 38
3.16.2 Usage . . . 38
3.16.3 Examples . . . 38
3.16.4 Notes . . . 38
3.17 *rtAshtech* . . . 39
3.17.1 Overview . . . 39
3.17.2 Usage . . . 39
3.17.3 Examples . . . 39
3.17.4 Notes . . . 39
3.18 *timeconvert* . . . 40
3.18.1 Overview . . . 40
3.18.2 Usage . . . 40
3.18.3 Examples . . . 40
3.18.4 Notes . . . 41
3.19 *WhereSat* . . . 42
3.19.1 Overview . . . 42
3.19.2 Usage . . . 42
3.19.3 Examples . . . 42
3.19.4 Notes . . . 43

The goal of the GPSTk project is to provide a world class, open source computing suite to the satellite navigation community. It is our hope that the GPSTk will empower its users to perform new research and create new applications.

GPS users employ practically every computational architecture and operating system. Therefore the design of the GPSTk suite is as platform-independent as possible. Platform independence is achieved through use of the ANSI-standard C++ programming language. The principles of object-oriented programming are used throughout the GPSTk code base in order to ensure that the code is modular, extensible and maintainable.

The GPSTk suite consists of a core ibrary and a set of applications. The library provides a wide array of functions that solve processing problems associated with GPS such as processing or using RINEX. The library is the basis for the more advanced applications distributed as part of the GPSTk suite.

The GPSTk is sponsored by Space and Geophysics Laboratory, within the Applied Research Laboratories at the University of Texas at Austin (ARL:UT). GPSTk is the by-product of GPS research conducted at ARL:UT since before the first satellite launched in 1978; it is the combined effort of many software engineers and scientists. In 2003 the research staff at ARL:UT decided to open source much of their basic GPS processing software as the GPSTk.

Part I

Theory

Chapter 1

The Global Positioning System in a Nutshell

The Global Positioning System is actually a U.S. 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 to 12 satellites are usually visible at a time.

1.1 GPS in a Nutshell

Each satellite broadcasts spread spectrum signals at 1575.42 and 1227.6 MHz, 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 just phase); it is also a range observation like the pseudorange, except (1) it has an unknown constant added to it (the phase ambiguity) and (2) it is much smoother (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.

1.1.1 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: 3 coordinates of position plus the receiver clock offset. The basic algorithm for the solution is described in the official GPS Interface Control Document, or 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 layers 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 directly computed if you have data on both the GPS frequencies. There is also a delay due to the troposphere, the lower part of the atmosphere. This delay too can be modeled and removed. There are many other errors 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 that are 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 explicitly model and remove errors. Creating new and robust models of phenomena that effects 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 Ephemeredes

The GPS position solution can be directly improved by using an improved satellite ephemeris. The U.S National Geospatial-Intelligence Agency (NGA) generates and makes publicly available a number of precise ephemeredes, which are more accurate satellite orbits. Satellite orbits described by the broadcast navigation message have an error on the order of meters; 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 ephemeredes possible.

1.2 GPS Data Sources

GPS observation data from many tracking stations are freely available 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; for example the Australian Regional GPS Network (ARGN) and global cooperatives such as NASA's Crust Dynamics Data Information System (CDIS).

1.2.1 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.

1.2.2 Receiver Protocols

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
```

The data here is a latitude, longitude fix at 51 deg 33.81 min North, 0 deg 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 utilize receiver-generated positions.

Closed standards are also common. SiRF is a proprietary protocol that is licensed to receiver manufacturers. Many receiver manufacturers implement their own binary protocols. While some of these protocols have been opened to the public, some have been reverse engineered.

Chapter 2

GPS File Formats

2.1 FIC

2.2 RINEX

Chapter 3

Converting Coordinates & Time

3.1 Transformations

Let $\mathbf{i}_x$, $\mathbf{i}_y$, $\mathbf{i}_z$ and $\mathbf{i}_\varepsilon$, $\mathbf{i}_\eta$, $\mathbf{i}_\zeta$ be two sets of orthagonal unit vectors

$$\mathbf{i}_\xi = l_1\mathbf{i}_x + m_1\mathbf{i}_y + n_1\mathbf{i}_z$$

$$\mathbf{i}_\eta = l_2\mathbf{i}_x + m_2\mathbf{i}_y + n_2\mathbf{i}_z$$

$$\mathbf{i}_\zeta = l_3\mathbf{i}_x + m_3\mathbf{i}_y + n_3\mathbf{i}_z$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \mathbf{R} \begin{bmatrix} \varepsilon \\ \eta \\ \zeta \end{bmatrix} \text{ or } \begin{bmatrix} \varepsilon \\ \eta \\ \zeta \end{bmatrix} = \mathbf{R^T} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} \mathbf{i}_x\cdot\mathbf{i}_\varepsilon & \mathbf{i}_x\cdot\mathbf{i}_\eta & \mathbf{i}_x\cdot\mathbf{i}_\zeta \\ \mathbf{i}_y\cdot\mathbf{i}_\varepsilon & \mathbf{i}_y\cdot\mathbf{i}_\eta & \mathbf{i}_y\cdot\mathbf{i}_\zeta \\ \mathbf{i}_z\cdot\mathbf{i}_\varepsilon & \mathbf{i}_z\cdot\mathbf{i}_\eta & \mathbf{i}_z\cdot\mathbf{i}_\zeta \end{bmatrix} = \begin{bmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \\ n_1 & n_2 & n_3 \end{bmatrix}$$

$$\mathbf{R^T} = \mathbf{R^{-1}}$$

Equations found here [1, pp. 81-82]

3.2 Time Systems

3.2.1 Solar & Sidereal Time

Since the beginning time has been kept by counting the the days. An apparent solar day is the minimum time elapsed between the sun crossing a specified meridian and then recrossing the same meridian. This form of time keeping is problematic because no two apparent solar days are of the same duration due to Earth's rotation around the sun as well as around its axis (the Earth does a little more than one rotation per apparent solar day). Also, Earth's rotational speed is not constant and its axis of rotation is tilted 23.5° to the orbital plane. These imperfections call for correction, and thus mean solar time was created. A day in mean solar time is defined as one revolution of a hypothetical sun that orbits at the equator, and is more commonly known as Greenwich Mean Time. Another solution is to base our day on the crossing of a star much farther away thus minimizing the effect of the Earth's orbital movement, this method of time keeping is known as sidereal time. A sidereal day is about 4 minutes shorter than a solar day, and is used heavily by astronomers. Sidereal time is not truly stable either so mean sidereal day was introduced, and is known as Greenwich Apparent Sidereal Time. Universal Time (UT) refers to any time scale based on the Earth's rotation. UT0 refers to the mean solar time at the prime meridian as obtained from astronomical observation, and UT1 is UT0 corrected for polar motion. Briefly ephemeris time was introduced to standardize the second, which was defined as 1/31556925.9747 of the year 1900. This was soon replaced by atomic time [4, pp. 84-86].

3.2.2 Atomic Time

The second is now defined by an atomic standard that is based on the resonance frequency of the cesium atom. To be precise, the second is defined as "9,192,631,770 periods of the radiation corresponding to the transition between the two hyperfine levels of the ground state of the cesium-133 atom," whose duration happens to exactly match the ephemeris second discussed in the previous section. The problem with detaching our time keeping method from the Earth is that as the Earth slows its rotation noon will move closer to midnight (over the duration of thousands of years, of course). Coordinated Universal Time (UTC) was introduced to prevent this. UTC is a compromise between the precision of atomic time and the groundedness of Earth based time keeping, it uses the atomic second but introduces leap seconds (positive or negative) when necessary to keep UTC within .9 seconds of UT1 [4, pp. 86-87].

3.2.3 Time Formats

We are used to dealing with months, days, years, hours, minutes, and seconds, but such a time format makes for difficult epoch calculations over long periods. To solve this problem Julian Date (JD) was introduced. JD consists of a day count (days since noon UT on January 1$^{\mathrm{st}}$ 4713 B.C.) and a fraction of the current day. This makes for easy time differencing, but the length of the date can become cumbersome and the fact that a new day starts at noon confusing. To make things even easier Modified Julian Date (MJD) was created whos origin is midnight November 17^{th}, 1858.

$$\text{MJD} = \text{JD} - 2400000.5$$

In order to make Julian Date useful we need an easy was to go between calendar dates and JD. *timeconvert* does this and more with ease. The equations to convert from calendar date to JD are

$$\text{JD} = \text{INT}[365.25y] + \text{INT}[30.6001(m+1)] + D + \text{UT}/24 + 1720981.5$$

$$\begin{array}{lll} y = Y - 1 & \text{and } m = M + 12 & \text{if } M \leq 2 \\ y = Y & \text{and } m = M & \text{if } M > 2 \end{array}$$

where M is the month, D is the day, Y is the year, and INT$[x]$ returns just the integer part of the a number. To go from JD to calendar date

$$a = \text{INT}[\text{JD} + 0.5]$$

$$b = a + 1537$$

$$c = \text{INT}[(b - 122.1)/365.25]$$

$$d = \text{INT}[365.25c]$$

$$e = \text{INT}[(b - d)/30.6001]$$

$$D = b - d - \text{INT}[30.6001e] + \text{FRAC}[\text{JD} + 0.5]$$

$$M = e - 1 - 12\text{INT}[e/14]$$

$$Y = c - 4715 - \text{INT}[(7 + M)/10]$$

where FRAC$[x]$ returns just the fractional part of a real number. MJD Conversion found here [4, p. 88]. All other date conversions were found here [2, pp. 36-37]

3.2.4 GPS Time

GPS Time (GPST) is a continuously running composite time kept by cesium and rubidium frequency standards aboard the satellites and at monitor stations. While there are no leap seconds in GPST as there are in UTC, it is steered to stay within 1 μs of UTC, that is the difference between GPST and UTC is an integer number of seconds plus a fraction of a μs. GPST is formatted in terms of GPS weeks and the number of seconds into the current week. Finding these values is done easily if the Julian Date is known.

$$\text{GPS WEEK} = \text{INT}[(\text{JD} - 2444244.5)/7]$$

$$\text{SOW} = \text{FRAC}[(\text{JD} - 2444244.5)/7] \times 604800$$

where INT$[x]$ returns the integer part of a real number, FRAC$[x]$ returns the fractional part, and SOW stands for Second of Week.

Other useful quantities such as Day of Week and Second of Day can be found using *timeconvert* or the following equations.

$$\mathrm{DOW} = \mathrm{modulo}\{\mathrm{INT}[\mathrm{JD} + 0.5], 7\}$$

$$\mathrm{SOD} = \mathrm{modulo}\{\mathrm{FRAC}[\mathrm{JD} + 0.5], 7\} \times 86400$$

where DOW=0 corresponds to Monday, DOW=1 corresponds to Tuesday, and so on.

JD and GPS Week equations were found here [2, pp. 36-37], SOD derived from DOW equation

3.2.5 Z-Count

Satellites keep internal time with Z-count, whose epoch period is 1.5 seconds (a convenient unit for communications timing). The full Z-count is 29 bits, the 10 bit GPS week folloed by a 19 bit Time of Week (TOW) expressed in Z-counts (or 1.5 second units). The truncated Z-count has a 17 bit TOW that is expressed in units of 6 seconds, or the length of one subframe's transmission time. Simply multiply the truncated TOW by 4 to get the full TOW [5, pp. 86-88].

$$\mathrm{TOW} = \mathrm{FRAC}[(\mathrm{JD} - 2444244.5)/7] \times 403200$$

$$\text{Truncated TOW} = \mathrm{FRAC}[(\mathrm{JD} - 2444244.5)/7] \times 100800$$

Equations derived from SOW equation above

3.3 Earth Fixed Coordinates

3.3.1 ECI to ECF

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_{ECF} = T^{xyz}_{XYZ} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{ECI}$$

$$T^{xyz}_{XYZ} = WSNP$$

P - applies precession, from epoch 2000.0 to the current time; N - applies nutation, from epoch 2000.0 to the current time; S - applies rotation to account for true sidereal time; W - applies polar motion;

Equations found on page 85 of Fundamentals of Orbit Determination paper book

3.3.2 WGS-84

The World Geodetic System 1984 (WGS-84) is fixed physical model of Earth produced by the Department of Defense to which many different reference frames can be attached. WGS-84 consists of two parts, a model of Earth's gravitational field, and an ellipsoid describing the Earth's general shape. When dealing with locations on the Earth's surface the ellipsoid provides the foundation for the geodetic coordinate system used by GPS. The ellipsoid's cross-sections parallel to the equatorial plane are circular while those orthogonal are elliptical. The ellipses are parameterized by an eccentricity e, a flattening f, and sometimes a second eccentricity e'

$$e = \sqrt{1 - \frac{b^2}{a^2}}$$

$$f = 1 - \frac{b}{a}$$

$$e' = \sqrt{\frac{a^2}{b^2} - 1} = \frac{a}{b}e$$

where a, the semimajor axis, is the value of the mean equatorial radius of Earth (6,378.137 km) and b, the semiminor axis, is the value of the polar radius of Earth (6,356.7523142 km) [3, pp. 25-26].

3.3.3 Coordinate Systems

Now that WGS-84 is defined it is important to understand what coordinate systems can be attached to the ellipsoid and how to move between these different systems. The GPS Toolkit comes with *poscvt*, an application that gives users the ability to easily convert coordinates in one reference frame to another. The coordinate systems that *poscvt* recognizes are Cartesian (or XYZ), geodetic, geocentric, and spherical coordinates. These systems and the formulas to convert between them are discussed below.

Cartesian (XYZ) Coordinates

The Earth Centered Earth Fixed (ECEF) Cartesian coordinate system is fixed to the WGS-84 ellipsoid and is the common ground that makes going between the Earth Centered Inertial (ECI) reference frame used by the satellites and the systems we are used to (such as latitude, longitude, and height) manageable. The equatorial plane makes the xy-plane with the $+x$-axis pointing toward $0°$ longitude and the $+y$-axis pointing toward $90°$ E longitude. The z-axis is normal to the equatorial plane and points to the geographical north pole. The conversion formulas presented in the next sections will convert to and from this Cartesian reference frame, and so to convert between two non-Cartesian coordinate systems the XYZ system will be used as an intermediary [3, p. 24].

Geodetic Coordinates

The geodetic coordinate parameters are longitude λ, latitude ϕ, and height h. Longitude is defined as the angle between the position and the x-axis in the equatorial plane, and is easily computed given a position in Cartesian coordinates. Let a user's position $\mathbf{U} = (x_u, y_u, z_u)$, then

$$\lambda = \begin{cases} \arctan\left(\frac{y_u}{x_u}\right), & x_u \geq 0 \\ 180^\circ + \arctan\left(\frac{y_u}{x_u}\right), & x_u < 0 \text{ and } y_u \geq 0 \\ -180^\circ + \arctan\left(\frac{y_u}{x_u}\right), & x_u < 0 \text{ and } y_u < 0 \end{cases}$$

where negative angles signal west longitude.

Latitude and height are not so straight forward. Latitude is determined by drawing a vector normal to the ellipsoid, beginning somewhere on the equatorial plane and terminating at the users position, we will call this the user vector. The smallest angle between this vector and the equatorial plane is the user's latitude, it is a North latitude for positive angles and South for negative. Notice that unless the user is at a pole or on the equator the vector does not pass through the center of the Earth. The users height is found by taking the magnitude of the vector originating on and normal to the ellipsoid and terminating at the user's position. Latitude ϕ and height h are found using the following equations

$$\phi = \arctan\left(\frac{z_u + e'^2 z_0}{r}\right)$$

$$h = U\left(1 - \frac{b^2}{aV}\right)$$

where

$$r = \sqrt{x_u^2 + y_u^2}$$

$$E^2 = a^2 - b^2$$

$$F = 54b^2 z_u^2$$

$$G = r^2 + (1 - e^2)z_u^2 - e^2 E^2$$

$$c = \frac{e^4 F r^2}{G^3}$$

$$s = \sqrt[3]{1 + c + \sqrt{c^2 + 2c}}$$

$$P = \frac{F}{3\left(s + \frac{1}{s} + 1\right)^2 G^2}$$

$$Q = \sqrt{1 + 2e^4 P}$$

$$
\begin{aligned}
r_0 = -\frac{Pe^2 r}{1+Q} + \sqrt{\frac{1}{2}a^2\left(1+\frac{1}{Q}\right) - \frac{P(1-e^2)z_u^2}{Q(1+Q)} - \frac{1}{2}Pr^2} \\
U = \sqrt{(r-e^2 r_0)^2 + z_u^2} \\
V = \sqrt{(r-e^2 r_0)^2 + (1-e^2)z_u^2} \\
z_0 = \frac{b^2 z_u}{aV}
\end{aligned}
$$

Going back to Cartesian coordinates from the geodetic system $(\lambda\ \phi\ h)$ can be done more compactly

$$
\mathbf{u} = \begin{bmatrix} \frac{a\cos\lambda}{\sqrt{1+(1-e^2)\tan^2\phi}} + h\cos\lambda\cos\phi \\ \frac{a\sin\lambda}{\sqrt{1+(1-e^2)\tan^2\phi}} + h\sin\lambda\cos\phi \\ \frac{a(1-e^2)\sin\phi}{\sqrt{1-e^2\sin^2\phi}} + h\sin\phi \end{bmatrix}
$$

where $\mathbf{u}$ is the user's position vector [3, 4, pp. 26-28, p. 76].

Geocentric Coordinates

$$
\begin{aligned}
x &= r\cos\phi\cos\lambda \\
y &= r\cos\phi\sin\lambda \\
z &= r\sin\phi
\end{aligned}
$$

where λ and ϕ are geocentric longitude and latitude
found on page 82 in the Fundamentals of Orbital Determination paper book

Spherical Coordinates

Topocentric Coordinates

$$
\mathbf{r}_t = T_t(\mathbf{r} - \mathbf{r}_s) = T_t\rho
$$

$\mathbf{r}$ and $\mathbf{r}_s$ are the position vectors of the observer and satellite respectively in the Earth-fixed system

$$
T_t = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\phi\cos\lambda & -\sin\phi\sin\lambda & \cos\phi \\ \cos\phi\cos\lambda & \cos\phi\sin\lambda & \sin\phi \end{bmatrix}
$$

where λ and ϕ are geocentric longitude and latitude
found on page 84 in the Fundamentals of Orbital Determination paper book
to find *azimuth* (Az) and *elivation* (El)

$$
\begin{aligned}
&\sin \mathrm{El} = \tfrac{z_t}{r_t} && -90^\circ \le \mathrm{El} \le 90^\circ \\
&\sin \mathrm{Az} = \tfrac{x_t}{r_{xy}} \\
&\cos \mathrm{Az} = \tfrac{y_t}{r_{xy}} && 0^\circ \le \mathrm{Az} \le 360^\circ
\end{aligned}
$$

Equations found on pages 84-85 in Fundamentals of Orbit Determination paper book

3.4 References

[1] Richard H. Battin. *An Introduction to the Mathematics and Methods of Astrodynamics*. AIAA Press, Reston, Virginia, revised edition, 1999.

[2] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *GPS: Theory and Practice*. Springer-Verlag Wien, New York, NY, 5th edition, 2001.

[3] Elliot D. Kaplan, editor. *Understanding GPS: Principles and Applications*. Artech House Publishers, 685 Canton Street, Norwood, MA, 1996.

[4] Pratap Misra and Per Enge. *Global Positioning Sytem: Signals, Measurements and Performance*. Ganga-Jamuna Press, Lincoln, Massachusetts, 2004.

[5] James Bao-Yen Tsui. *Fundamentals of Global Positioning System Receivers: A Software Approach*. John Wiley & Sons, New York, 2000.

Part II

Usage, Examples & Notes

	Tool	Description	Execution Example
Transforms	calgps	generates a GPS calendar	`calgps -Y 2004`
	poscvt	converts a given input position to other position formats	`poscvt --geodetic="30.28 262.26700 167.64"`
	timeconvert	converts given input time to other time formats	`timeconvert --calendar="07 04 2006"`
	WhereSat	outputs expected location of a satellite	`WhereSat -b arl2100.06n -p 3`
Collecting & Converting	rtAshtech	records observations from an Ashtech receiver	`rtAshtech -p /dev/ttyS1 -o "minute%03j%02H%02m.%06yo"`
	ficfica ficafic fic2rin	convert fic files between ASCII, binary, and Rinex formats	`fic2rin fic2100.06 rin121.06n`
	mdp2fic mdp2rinex	convert MDP files to fic or rinex files	`mdp2rinex -i mdpfile -o arl2100.06o`
	novaRinex	convert Novatel files to Rinex files	`novaRinex --input nova2100.06 --obstype L1`
	navdmp	dumps information from nav files to human readable formats	`navdmp -i arl2100.06n -o arl2100.06.dmp`
	RinexDump	dumps observation data for specified satellites from a Rinex file	`RinexDump arl2100.06o 3 4 L1 L2`
Comparing & Validating	ephdiff	compares the satellite positions from two ephemeris sources	`ephdiff arl2100.06n fic2100.06`
	ficdiff	compares contents of two FIC files	`ficidff fic12100.06 fic22100.06`
	ficcheck ficacheck	reads a FIC file and checks it for errors reporting the first found	`ficcheck fic2100.06 -t "07/20/2006 11:00:00"`
	row/rnw/rmwdiff	compares contents of two RINEX files	`rowdiff arl1210.06o arl22100.06o`
	row/rnw/rmwcheck	read Rinex files and checks it for errors reporting the first found	`rnwcheck arl210.06n -e "07/20/2006 11:00:00"`
	navsum RinSum	summarizes the contents of nav/Rinex files	`RinSum -i arl2100.06o --EpochBeg 2006,07,20,13,20,00`
	mdptool	manipulates MDP data streams	`mdptool -i mdpfile --pvt --obs`
	reszilla	computes various residuals from GPS data	`reszilla -o arl210.06o -e arl2100.06n`
Editing Data	mergeFIC	sorts and merges input FIC files into a single file	`mergeFIC -i fic12100.06 -i fic22100.06 -o ficmerge2100.06`
	mergeRinObs/Nav/Met	sorts and merges RINEX files	`mergeRinNav -i arl2100.06n -i arl2110.06n arl210-211.06n`
	NavMerge	merges Rinex nav files into a single file	`NavMerge -oarlnavs.06n arl2100.06n arl2110.06n`
	rinexthin	decimates an input Rinex observation files to desired data rate	`rinexthin -f arl2100.06o -s 30 -o arl2100thin.06n`
	ResCor	edits Rinex files and computes corrections	`ResCor -IFalr2100.06o -OFarl2100mod.06o -DS12,12:00:00`
	DiscFix	cycle slip corrector	`DiscFix -i arl2100.06o --DT 1.5`
Iono	IonoBias	solves interfrequency biases and a simple ionosphere model	`IonoBias --input arl2100.06o --nav arl2100.06n --XSat 3`
	TECMaps	creates maps of Total Electron Content (TEC)	`TECMaps --input arl2100.06o --nav arl2100.06n --LinearFit`
Positioning	PRSolve	generates autonomous position solution	`PRSolve -o alr2100.06o -n arl2100.06nn --XPRN 12`
	rinexpvt	generates autonous position solution	`rinexpvt -o alr2100.06o -n arl2100.06n`
	DDBase	computes a network solution using carrier phase	`DDBase ...  --ObsFile arl2100.06o --PosXYZ x,y,z,1 --Fix`
	vecsol	estimates short baseline using range or carrier phase	`vecsol station12100.06o station22100.06o`

Table 3.0: GPSTk Applications at a Glance

3.5 *calgps*

3.5.1 Overview

This application generates a dual GPS and Julian calendar. The arguments and format are inspired by the UNIX 'cal' utility. With no arguments, the current argument is printed. The last and next month can also be printed. Also, the current or any given year can be printed.

3.5.2 Usage

Optional Arguments

Short Arg.	Long Arg.	Description
-h	–help	Generates help output.
-3	–three-months	Prints a GPS calendar for the previous, current, and next month.
-y	–year	Prints a GPS calendar for the entire current year.
-Y year	–specific-year=NUM	Prints a GPS calendar for the entire specified year.

3.5.3 Examples

```
> calgps -3


                    Jul 2006
1381                                       1-182
1382  2-183  3-184  4-185  5-186  6-187  7-188  8-189
1383  9-190 10-191 11-192 12-193 13-194 14-195 15-196
1384 16-197 17-198 18-199 19-200 20-201 21-202 22-203
1385 23-204 24-205 25-206 26-207 27-208 28-209 29-210
1386 30-211 31-212

                    Aug 2006
1386               1-213  2-214  3-215  4-216  5-217
1387  6-218  7-219  8-220  9-221 10-222 11-223 12-224
1388 13-225 14-226 15-227 16-228 17-229 18-230 19-231
1389 20-232 21-233 22-234 23-235 24-236 25-237 26-238
1390 27-239 28-240 29-241 30-242 31-243
 . . .
```

```
> calgps -Y 1998


                    Jan 1998
 938                                1-001  2-002  3-003
 939  4-004  5-005  6-006  7-007  8-008  9-009 10-010
 940 11-011 12-012 13-013 14-014 15-015 16-016 17-017
 941 18-018 19-019 20-020 21-021 22-022 23-023 24-024
 942 25-025 26-026 27-027 28-028 29-029 30-030 31-031

                    Feb 1998
 943  1-032  2-033  3-034  4-035  5-036  6-037  7-038
 944  8-039  9-040 10-041 11-042 12-043 13-044 14-045
 945 15-046 16-047 17-048 18-049 19-050 20-051 21-052
```

```
946  22-053 23-054 24-055 25-056 26-057 27-058 28-059

                   Mar 1998
. . .
```

3.5.4 Notes

If multiple options are given only the first is considered.

3.6 *ficafic ficfica*

3.6.1 Overview

These applications convert navigation message data between variations of the FICformat, a format for GPS observations established by ARL:UT.

3.6.2 Usage

```
ficafic usage: ficafic <input fica file> <output fic file name>
ficfica usage: ficfica <input fic file> <output fica file name>
```

3.6.3 Examples

```
> ficfica fic06.187 fica06.187
```

```
File Snippets

Binary FIC File

0000000
*
0000020                                    B   L   K       m  \0  \0  \0
0000030  \0  \0  \0  \0      \0  \0  \0  \0  \0  \0  \0   f 005  \0  \0
0000040 022  \0  \0  \0   >   f 301   " 260   i   {   !   f  \0   d 026
0000050 335 344   8  \t 002   b   C 035 205   7   4 027 241 372 210 006
0000060 006   }   Y   / 301 374   ?  \0   \   S 021   8   >   f 301   "
 . . .

ASCII FIC File

BLK    109    0   32    0
        1382          18   583099966   561736112   375652454   154723549
   490955266   389298053   109640353   794393862     4193473   940659548
   583099966   561744492   792779231   218793822   800301952    12009725
   793943984    14182503    56922219   427630416   583099966   561753060
  1073203199   309077037     1329639    15188054   182084772   733918588
  1072216082   792738524
BLK      9   60    0    0
 .13900000000000D+03 .35800000000000D+03 .41142600000000D+06 .10000000000000D+01
 .10000000000000D+01 .13820000000000D+04 .10000000000000D+01 .00000000000000D+00
 .00000000000000D+00 .91136000000000D+06 .00000000000000D+00-.10244548320770D-07
 .41760000000000D+06 .00000000000000D+00-.14779288903810D-11-.24207541719079D-03
 .00000000000000D+00 .00000000000000D+00 .00000000000000D+00 .18000000000000D+02
 . . .
```

3.6.4 Notes

3.7 *fic2rin*

3.7.1 Overview

This application converts navigation messages between the FIC format, a format for GPS observations established by ARL:UT, and the RINEX format.

3.7.2 Usage

```
fic2rin usage: fic2rin <input FIC file> <output RINEX file name>
```

3.7.3 Examples

```
> fic2rin fic06.187 rin1870.06
```

```
File Snippets

Binary FIC File

0000000
*
0000020                                    B   L   K       m  \0  \0  \0
0000030  \0  \0  \0  \0      \0  \0  \0  \0  \0  \0  \0   f 005  \0  \0
0000040 022  \0  \0  \0   >   f 301   " 260   i   {   !   f  \0   d 026
0000050 335 344   8  \t 002   b   C 035 205   7   4 027 241 372 210 006
0000060 006   }   Y   / 301 374   ?  \0   \   S 021   8   >   f 301   "
 . . .

RINEX NAV File

     2.10           NAVIGATION                             RINEX VERSION / TYPE
fic2rin                                07/13/2006 11:48:58 PGM / RUN BY / DATE
                                                           END OF HEADER
 5 06  7  6 19 59 44.0  .199091155082D-03  .356976670446D-10  .000000000000D+00
     .118000000000D+03 -.656250000000D+00  .538879589355D-08  .997594152841D+00
    -.409781932831D-07  .710751442239D-02  .655464828014D-05  .515355578804D+04
     .417584000000D+06 -.104308128357D-06 -.249936238139D+01  .707805156708D-07
     .938194464982D+00  .241750000000D+03  .105751234129D+01 -.843570852398D-08
     .600024993449D-10  .100000000000D+01  .138200000000D+04  .000000000000D+00
     .240000000000D+01  .000000000000D+00 -.419095158577D-08  .118000000000D+03
     .411426000000D+06  .400000000000D+01
 . . .
```

3.7.4 Notes

3.8 *mdp2fic mdp2rinex*

3.8.1 Overview

The applications convert a variety of GPS related observations from the MDP format to FIC and RINEX formats. MDP is a format for network receiver interfaces derived by ARL:UT that can be used to serve observations over networks.

3.8.2 Usage

mdp2fic

Required Arguments

Short Arg.	Long Arg.	Description
-i	–mdp-input=ARG	Filename to read MDP data from. The filename of '-' means to use stdin.
-n	–nav=ARG	Filename to which FIC nav data will be written.

Optional Arguments

Short Arg.	Long Arg.	Description
-d	–debug	Increase debug level
-v	–verbose	Increase verbosity
-h	–help	Print help usage
-l	–log=ARG	Filename for (optional) output log file

mdp2rinex

Required Arguments

Short Arg.	Long Arg.	Description
-i	–mdp-input=ARG	Filename to read MDP data from. The filename of '-' means to use stdin.
-n	–obs=ARG	Filename to write RINEX obs data to. The filename of '-' means to use stdout.

Optional Arguments

Short Arg.	Long Arg.	Description
-d	–debug	Increase debug level
-v	–verbose	Increase verbosity
-h	–help	Print help usage
-n	–nav=ARG	Filename to write RINEX nav data to.
-t	–thinning=ARG	A thinning factor for the data, specified in seconds between points. Default: none.
-c	–12c=ARG	Enable output of L2C data in C2.

3.8.3 Examples

```
> mdp2fic -i mdp183.06 -o fic183.06 -l mdp2ficlog183.06
```

```
> mdp2rinex -i mdp183.06 -o rin183.06o -n rin183.06n -t 60
```

3.8.4 Notes

3.9 *mergeRinObs mergeRinNav mergeRinMet mergeFic*

3.9.1 Overview

The applications merge multiple RINEX observation, navigation, or meterolgical data files into a single coherent RINEX obs/nav/met file.

3.9.2 Usage

mergeRinObs

Required Arguments		
Short Arg.	Long Arg.	Description
-i	–input=ARG	An input RINEX Obs file, can be repeated as many times as needed.
-o	–output=ARG	Name for the merged output RINEX Obs file. Any existing file with that name will be overwritten.
Optional Arguments		
Short Arg.	Long Arg.	Description
-d	–debug	Increase debug level
-v	–verbose	Increase verbosity
-h	–help	Print help usage

mergeRinNav and *mergeRinNav* have the same usage.

3.9.3 Examples

```
> mergeRinObs -i arl280.06o -i arl2810.06o -o arl280-10.06o
```

```
> mergeRinNav -i arl280.06n -i arl2810.06n -o arl280-10.06n
```

```
> mergeRinMet -i arl280.06m -i arl2810.06m -o arl280-10.06m
```

3.9.4 Notes

3.10 *navdmp*

3.10.1 Overview

The application prints the contents of an FIC or RINEX file into a human readable file and allows filtering of the data.

3.10.2 Usage

navdmp

Required Arguments

Short Arg.	Long Arg.	Description
-i	–input=ARG	Name of an input navigation message file
-o	–output=ARG	Name of an output file

Optional Arguments

Short Arg.	Long Arg.	Description
-d	–debug	Increase debug level
-v	–verbose	Increase verbosity
-h	–help	Print help usage
-a	–all-records	Unless otherwise specified, use default values for record filtration.
-t	–time=TIME	Start time (of data) for processing
-e	–end-time=TIME	End time (of data) for processing
-p	–prn=NUM	PRN(s) to include
-b	–block=NUM	FIC block number(s) to process ((9)109 (Engineering) ephemerides, (62)162 (engineering) almanacs)
-r	–RINEX	Assume input file is a RINEX navigation message file

3.10.3 Examples

```
> bash-3.00$ navdmp -i algo1720.06n -o summary !!!!!WHAT ON EARTH-t "06/25/2006 10:30:00"!!!!! -p 1 -p 2 -r

Current filtering options:
       Start time:   01/10/0006 16:09:24
       End time:     01/01/4713 00:00:00
       PRNs:         1 2

Choose an option by number then push enter:
       1) Change the start time
       2) Change the end time
       3) Select specific PRNs
       5) Process the file
use ctrl-c to exit
? 5

processing...
```

```
Summary File Snippet

****************************************************************************
```

```
Broadcast Ephemeris (Engineering Units)

PRN :  2

              Week(10bt)     SOW     DOW   UTD     SOD   MM/DD/YYYY   HH:MM:SS
Clock Epoch:  1380( 356)  259200   Wed-3   172       0   06/21/2006   00:00:00
Eph Epoch:    1380( 356)  259200   Wed-3   172       0   06/21/2006   00:00:00
Transmit Week:1380
Fit interval flag :  0

          SUBFRAME OVERHEAD

               SOW    DOW:HH:MM:SS     IOD    ALERT   A-S
SF1 HOW:    259140  Tue-2:23:59:00   0x1C7      0     off
SF2 HOW:    259140  Tue-2:23:59:00    0xC7      0     off
SF3 HOW:    259140  Tue-2:23:59:00    0xC7      0     off

           CLOCK

Bias T0:       6.67711720E-06 sec
Drift:         3.29691829E-12 sec/sec
Drift rate:    0.00000000E+00 sec/(sec**2)
Group delay:  -1.72294676E-08 sec

           ORBIT PARAMETERS

Semi-major axis:         5.15369497E+03 m**.5
Motion correction:       4.82591530E-09 rad/sec
Eccentricity:            8.99635826E-03
Arg of perigee:          2.08978447E+00 rad
Mean anomaly at epoch:   3.30690945E-01 rad
Right ascension:        -7.28361281E-02 rad     -8.46642409E-09 rad/sec
Inclination:             9.50302779E-01 rad      8.85751181E-11 rad/sec

           HARMONIC CORRECTIONS

Radial        Sine:   1.09656250E+02 m    Cosine:   2.53281250E+02 m
Inclination   Sine:  -2.06753612E-07 rad  Cosine:   5.02914190E-08 rad
In-track      Sine:   6.54533505E-06 rad  Cosine:   5.60097396E-06 rad

           SV STATUS

Health bits:   0x00      URA index:    1
Code on L2:   reserved     L2 P Nav data:          on

***************************************************************************
```

3.10.4 Notes

3.11 *poscvt*

3.11.1 Overview

This application allows the user to convert among different coordinate system on the command line. Coordinate systems handled include Cartesian, geocentric, and geodetic.

3.11.2 Usage

Optional Arguments

Short Arg.	Long Arg.	Description
-d	–debug	Increase debug level
-v	–verbose	Increase verbosity
-h	–help	Print help usage
	–ecef=POSITION	ECEF “X Y Z” in meters
	–geodetic=POSITION	Geodetic “lat lon alt” in deg, deg, meters
	–geocentric=POSITION	Geocentric “lat lon radius” in deg, deg, meters
	–spherical=POSITION	Spherical “theta, pi, radius” in deg, deg, meters
-l	–list-formats	List the available format codes for use by the input and output format options.
-F	–output-format=ARG	Write the position with the given format

3.11.3 Examples

```
> poscvt --ecef="-4346070.69263 4561978.26297 803.498856837"


   ECEF (x,y,z) in meters           -4346070.6926 4561978.2630 803.4989
   Geodetic (llh) in deg, deg, m    0.00735641 133.61157352 -77345.2412
   Geocentric (llr) in deg, deg, m  0.00730656 133.61157352 6300791.7584
   Spherical (tpr) in deg, deg, m   89.99269344 133.61157352 6300791.7584
```

```
> poscvt -l

  %X %Y %Z (cartesian or ECEF in kilometers)
 %x %y %z (cartesian or ECEF in meters)
 %a %l %r (geocentric lat,lon,radius, longitude E, radius in meters)
 %A %L %h (geodetic lat,lon,height, longitude E, height in meters)
 %a %w %R (geocentric lat,lon,radius, longitude W, radius in kilometers)
 %A %W %H (geodetic lat,lon,height, longitude W, height in kilometers)
 %t %p %r (spherical theta, phi, radius, degrees and meters)
 %T %P %R (spherical theta, phi, radius, radians and kilometers)
```

```
> poscvt --ecef="-4346070.69263 4561978.26297 803.498856837" -F "\%A \%L \%h"

\0.007356 \133.611574 \-77345.241247
```

3.11.4 Notes

3.12 *PRSolve*

3.12.1 Overview

The application reads one or more Rinex observation files, plus one or more navigation (ephemeris) files, and computes an autonomous pseudorange position solution, using a RAIM-like algorithm to eliminate outliers. Output is to the log file, and also optionally to a Rinex obs file with the position solutions in auxiliary header blocks.

3.12.2 Usage

navdmp

Required Arguments		
Short Arg.	Long Arg.	Description
-o	–obs	Input Rinex observation file(s)
-n	–nav	Input navigation (ephemeris) file(s) (Rinex or SP3)
Optional Arguments: Input		
-f		File containing more options
	–obsdir	Directory of input observation file(s)
	–navdir	Directory of input navigation file(s)
	–decimate	Decimate data to time interval dt
	–EpochBeg	Start time, arg is of the form YYYY,MM,DD,HH,Min,Sec
	–GPSBeg	Start time, arg is of the form GPSweek,GPSsow
	–EpochEnd	End time, arg is of the form YYYY,MM,DD,HH,Min,Sec
	–GPSEnd	End time, arg is of the form GPSweek,GPSsow
	–CA	Use C/A code pseudorange if P1 is not available
Optional Arguments: Configuration		
	–RMSlimit	Upper limit on RMS post-fit residuals (m) for a good solution
	–SlopeLimit	Upper limit on RAIM 'slope' for a good solution
	–Algebra	Use algebraic algorithm (otherwise linearized LS)
	–DistanceCriterion	Use distance from a priori as convergence criterion (else RMS)
	–ReturnAtOnce	Return as soon as a good solution is found
	–NReject	Maximum number of satellites to reject
	–NIter	Maximum iteration count (linearized LS algorithm)
	–Conv	Minimum convergence criterion (m) (LLS algorithm)
	–MinElev	Minimum elevation angle (deg) (only if –PosXYZ)
	–XPRN	Exclude this satellite.
	–Trop <model,T,P,H>	Trop model (one of BL,SA,NB,GG,GGH (cf.GPSTk)), with OPTIONAL weather Temp(C),Press(mb),RH(%)
Optional Arguments: Output		
	–Log	Output log file name (prs.log).
	–PosXYZ <X,Y,Z>	Known position (ECEF,m), used to compute output residuals.

	–APSout	Output autonomous pseudorange solution (APS - no RAIM)
	–TimeFormat	Output time format (ala DayTime) (default: %4F %10.3g)

Optional Arguments: RINEX Output

	–RinexFile	Output Rinex obs file name
	–RunBy	Output Rinex header 'RUN BY' string
	–Observer	Output Rinex header 'OBSERVER' string
	–Agency	Output Rinex header 'AGENCY' string
	–Marker	Output Rinex header 'MARKER' string
	–Number	Output Rinex header 'NUMBER' string

Optional Arguments: Help

	–verbose	Print extended output.
	–debug	Print very extended output.
-h	–help	Print syntax and quit.

3.12.3 Examples

```
> PRSolve -o arl2800.06o -n arl2800.06n

PRSolve, part of the GPSTK ToolKit, Ver 1.7 3/06, Run 2006/08/07 11:57:46
Opened log file prs.log
Weighted average RAIM solution for all files
   918129.320229 -4346071.108765 4561977.869659
Covariance of RAIM solution for all files
        0.000150      -0.000061       0.000058
       -0.000061       0.000427      -0.000248
        0.000058      -0.000248       0.000493
PRSolve timing: 7.720 seconds.
```

```
> PRSolve -o arl2800.06o -n arl2800.06n --EpochBeg 2006,1,1,00,00,00 --EpochEnd 2006,1,1,12,00,00

PRSolve, part of the GPSTK ToolKit, Ver 1.7 3/06, Run 2006/08/07 11:58:06
Opened log file prs.log
Weighted average RAIM solution for all files
   918129.968984 -4346071.600388 4561978.175321
Covariance of RAIM solution for all files
        0.000315      -0.000130       0.000155
       -0.000130       0.000918      -0.000516
        0.000155      -0.000516       0.001041
PRSolve timing: 3.870 seconds.
```

3.12.4 Notes

3.13 *reszilla*

3.13.1 Overview

3.13.2 Usage

Required Arguments

Short Arg.	Long Arg.	Description
-o	–obs1=ARG	Observation data file name. If this option is specified more than once the contents of all files will be used.

Optional Arguments

Short Arg.	Long Arg.	Description
-h	–help	Generates help and usage.
-2	–position=ARG	Second receiver's observation data file name. Only used when computing a double difference. If this option is specified more than once the contents of all the files will be used.
	–msc=ARG	Station coordinate file.
-e	–ephemeris=ARG	Ephemeris data file name (either broadcast in RINEX nav, broadcast in FIC, or precise in SP3).
-w	–weather	Weather data file name (RINEX met format only).
-n	–search-near	Use BCEphemeris.searchNear()
-c	–clock-from-rinex	Use the receiver clock offset from the rinex obs data.
	–svtime	Observation data is in SV time frame. The default is RX time frame.
	–check-obs	Report data rate, order of data, data present, data gaps.
	–keep-unhealthy	Keeps unhealthy SVs in the statistics, default is to toss.
-s	–no-stats	Don't compute & output the statistics.
	–cycle-slips	Output a list of cycle slips.
-r	–raw-output=ARG	Dump the computed residuals/ords into specified file. If '-' is given as the file name, the output is sent to standard output. The default is to not otput the raw residuals.
	–start-time=TIME	Ignore obs data prior to this time in the analysis. The time is specified using the format %4Y/%03j/%02H:%02M:%05.2f. The default value is to start with the first data found.
	–stop-time=TIME	Ignore obs data after to this time in the analysis. The time is specified using the format %4Y/%03j/%02H:%02M:%05.2f. The default value is to process all data.
-t	–time-format=ARG	Daytime format specifier used for the timestamps in the raw output. The default is "%Y %3j %02H:%02M:%04.1f". If this option is specified with the format as "s", the format "%Y %3j %7.1s" is used. If this option is specified with the format as "s", the format "%Y %3j %02H:%02M:%02S" is used.
	–omode=ARG	ORD mode: P1P2, C1P2, C1, P1, P2. The default is p1p2

	–clock-est	Compute a linear clock estimate.
	–ddmode=ARG	Double difference residual mode: none, sv, or c1p2. The default is sv.
	–min-arc-time=ARG	The minimum length of time (in seconds) that a sequence of observations must span to be considered as an arc. The default value is 60.0 seconds.
	–min-arc-gap=ARG	The minimum length of time (in seconds) between two arcs for them to be considered separate arcs. The default value is 60.0 seconds.
	–min-arc-length=NUM	The minimum number of epochs that can be considered an arc. The default value is 5 epochs.
-b	–elev-bin=ARG	A range of elevations, used in computing the statistical summaries. Repeat to specify multiple bins. The default is "-b 0-10 -b 10-20 -b 20-60 -b 10-90".
	–sigma=NUM	Multiplier for sigma stripping used in computation of statistics on the raw residuals. The default value is 6.
-v	–verbosity=NUM	How much detail to provide about intermediate steps.
	0	nothing but the results
	1	Output status before potentially time consuming operations (default)
	2	more details about each step and the options chosen
	3	add the reasons for editing data
	4	dump intermediate values for each epoch (can be QUITE verbose)

```
Types in the raw output files:
   0 - c1p2 observed range deviation
   50 - computed clock, difference from estimate, strip
   51 - linear clock estimate, abdev
Double difference types:
   10 - c1     20 - c2
   11 - p1     21 - p2
   12 - l1     22 - l2
   13 - d1     23 - d2
   14 - s1     24 - s2
```

3.13.3 Examples

```
reszilla --omode=p1 --svtime --msc=mscoords.cfg -m 85401 -o asm2004.138 -e s011138a.04n
```

3.13.4 Notes

The criteria min-arc-time and min-arc-length are both required to be met for a arc to be valid in double difference mode. All output quantities (stddev, min, max, ord, clock, double differnce, ...) are in meters.

3.14 *RinexDump*

3.14.1 Overview

The application reads a RINEX file and dumps the obervation data for the given satellite(s) to the standard output.

3.14.2 Usage

```
RinexDump usage: RinexDump [-n] <$rinex file$> [<satellite(s)> <$obstype(s)>]

The optional argument -n tells RinexDump its output should be purely numeric.
```

3.14.3 Examples

```
> RinexDump algo1580.06o 3 4 5

# Rinexdump file: algo1580.06o Satellites: G03 G04 G05 Observations: ALL
# Week GPS_sow Sat          L1 L S          L2 L S          C1 L S
1378 259200.000 G03 -3843024.647 0 3 -2994560.443 0 1 23796436.087 0 0
1378 259230.000 G03 -3954052.735 0 3 -3081075.654 0 2 23775308.750 0 0
1378 259260.000 G03 -4064994.465 0 2 -3167523.561 0 3 23754197.617 0 0
1378 259290.000 G03 -4175846.973 0 3 -3253901.944 0 3 23733104.211 0 0
1378 259320.000 G03 -4286607.460 0 4 -3340208.647 0 3 23712026.249 0 0
1378 259350.000 G03 -4397272.869 0 4 -3426441.227 0 3 23690967.159 0 0

 . . .

         P2 L S          P1 L S          S1 L S          S2 L S
23796439.457 0 0 23796436.350 0 0     21.100 0 0      11.000 0 0
23775311.168 0 0 23775308.182 0 0     22.100 0 0      17.800 0 0
23754199.648 0 0 23754196.550 0 0     17.000 0 0      18.600 0 0
23733104.928 0 0 23733102.480 0 0     19.900 0 0      21.600 0 0
23712027.682 0 0 23712024.790 0 0     24.200 0 0      19.300 0 0
23690968.861 0 0 23690965.837 0 0     25.600 0 0      19.900 0 0

 . . .
```

3.14.4 Notes

MATLAB and Octave can read the purely numeric output.

3.15 *rinexpvt*

3.15.1 Overview

The application generates a user position based on RINEX observation data with the option of including navigation and meteriological data to aid error correction.

3.15.2 Usage

navdmp

Required Arguments

Short Arg.	Long Arg.	Description
-o	–obs-file=ARG	RINEX obs file

Optional Arguments

Short Arg.	Long Arg.	Description
-d	–debug	Increase debug level
-v	–verbose	Increase verbosity
-h	–help	Print help usage
-n	–nav-file=ARG	RINEX Nav file. Required for single frequency ionosphere correction.
-p	–pe-file=ARG	SP3 Precise Ephemeris File. Repeat this for each input file.
-m	–met-file=ARG	RINEX Met File
-t	–time-format=ARG	Alternate time format string.
-e	–enu=ARG	Use the following as origin to solve for East/North/Up coordinates, formatted as a string: "X Y Z"
-l	–elevation-mask=ARG	Elevation mask (degrees)
-s	–single-frequency	Use only C1 (SPS)
-f	–dual-frequency	Use only P1 and P2 (PPS)
-i	–no-ionosphere	Do NOT correct for ionosphere delay.
-x	–no-closest-ephemeris	Allow ephemeris use outside of fit interval.
-c	–no-carrier-smoothing	Do NOT use carrier phase smoothing.

3.15.3 Examples

```
> rinexpvt -o arl2800.06o -n arl2800.06n

2006 1 1 00 00 0.000000 918128.1413 -4346066.38713 4561976.84865 322.333995519
2006 1 1 00 00 30.000000 918128.209212 -4346067.60732 4561976.93485 323.041856353
2006 1 1 00 01 0.000000 918128.302764 -4346068.04452 4561977.21068 323.429649855
2006 1 1 00 01 30.000000 918128.391428 -4346068.3532 4561977.38928 323.717577661
2006 1 1 00 02 0.000000 918128.50273 -4346068.53469 4561977.48638 323.86573351
2006 1 1 00 02 30.000000 918128.529272 -4346068.41506 4561977.46288 323.78986994
2006 1 1 00 03 0.000000 918128.646582 -4346068.55693 4561977.52889 323.955585289
2006 1 1 00 03 30.000000 918128.740209 -4346068.77352 4561977.6377 324.13232439
2006 1 1 00 04 0.000000 918128.739294 -4346068.83903 4561977.68601 324.180075896
2006 1 1 00 04 30.000000 918128.781829 -4346068.85625 4561977.77165 324.239920157
2006 1 1 00 05 0.000000 918128.861036 -4346069.05268 4561977.91535 324.454651606
2006 1 1 00 05 30.000000 918128.933265 -4346069.40007 4561978.12808 324.786489416
2006 1 1 00 06 0.000000 918128.950514 -4346069.25246 4561978.14827 324.733986098
```

```
2006 1 1 00 06 30.000000 918128.960248 -4346069.24879 4561978.11298 324.748810797
2006 1 1 00 07 0.000000 918128.976853 -4346069.3422 4561978.17787 324.858597826
 . . .
```

```
> rinexpvt -o arl2800.06o -n arl2800.06n -m arl2800.06m

2006 1 1 00 00 0.000000 918128.1413 -4346066.38713 4561976.84865 322.333995519
2006 1 1 00 00 30.000000 918128.209212 -4346067.60732 4561976.93485 323.041856353
2006 1 1 00 01 0.000000 918128.401075 -4346068.40185 4561977.50754 323.99086869
2006 1 1 00 01 30.000000 918128.488498 -4346068.70699 4561977.68361 324.275285634
2006 1 1 00 02 0.000000 918128.598571 -4346068.88502 4561977.77824 324.42000745
2006 1 1 00 02 30.000000 918128.623895 -4346068.76203 4561977.75232 324.340785521
2006 1 1 00 03 0.000000 918128.739997 -4346068.90062 4561977.81596 324.503217171
2006 1 1 00 03 30.000000 918128.832428 -4346069.114 4561977.92245 324.676746145
2006 1 1 00 04 0.000000 918128.830326 -4346069.1764 4561977.9685 324.721360094
2006 1 1 00 04 30.000000 918128.871684 -4346069.19058 4561978.05191 324.778138464
2006 1 1 00 05 0.000000 918128.949723 -4346069.38404 4561978.19345 324.989874831
2006 1 1 00 05 30.000000 918129.020728 -4346069.7283 4561978.40383 325.318381098
2006 1 1 00 06 0.000000 918129.036829 -4346069.57789 4561978.42195 325.263023987
2006 1 1 00 06 30.000000 918129.045424 -4346069.57149 4561978.38464 325.275063272
2006 1 1 00 07 0.000000 918129.0609 -4346069.66224 4561978.44755 325.382132551
 . . .
```

3.15.4 Notes

Though not stated in the required options lists either a RINEX navigation file or an SP3 Precise Ephemeris File is needed, using the -n or -p option respectively. When using precise ephemeris 3 files must be included, the previous day's, the current day's and the next day's.

3.16 *rinexthin*

3.16.1 Overview

This application decimates an input RINEX observation file to a specified data rate.

3.16.2 Usage

navdmp

Required Arguments

Short Arg.	Long Arg.	Description
-f	–filename=ARG	RINEX obs file to be thinned.
-s	–Seconds=NUM	The desired data rate.
-o	–filename=ARG	RINEX obs file with thinned obs.

3.16.3 Examples

```
> rinexthin -f arl2800.06o -s 60 -o arl2800thin.06o

Obs read:   2880
Obs written: 1440
```

3.16.4 Notes

3.17 *rtAshtech*

3.17.1 Overview

This application logs observations from an Ashtech Z-XII receiver. It records observations directly into the RINEX format. A number of optional outputs are possible. The raw messages from a receiver can be recorded. Observations can also be recorded in a format that is easily imported into numerical packages.

3.17.2 Usage

rtAshtech

Optional Arguments

Short Arg.	Long Arg.	Description
-h	–help	Print help usage
-v	–verbose	Increased diagnostic messages
-r	–raw	Record raw observations
-l	–log	Record log entries
-t	–text	Record observations as simple text files
-p	–port=ARG	Serial port to use
-o	–rinex-obs=ARG	Naming convention for RINEX obs files
-n	–rinex-nav=ARG	Naming convention for RINEX nav message files
-T	–text-obs=ARG	Naming convention for obs in simple text files

3.17.3 Examples

```
> rtAshtech -p /dev/ttyS1
```

```
> rtAshtech -o "minute\%03j\%02H\%02M.\%02yo"
```

3.17.4 Notes

Only works on UNIX systems with POSIX compliant serial ports.

3.18 *timeconvert*

3.18.1 Overview

This application allows the user to convert among time formats associated with GPS. Time formats include: civilian time, Julian day of year and year, GPS week and seconds of week, Z counts, and Modified Julian Date (MJD).

3.18.2 Usage

Optional Arguments

Short Arg.	Long Arg.	Description
-d	–debug	Increase debug level
-v	–verbose	Increase verbosity
-h	–help	Print help usage
-c	–calendar=TIME	“Month(numeric) DayOfMonth Year”
-r	–rinex=TIME	”Month(numeric) DayOfMonth Year Hour:Minute:Second”
-R	–rinex-file=TIME	”Year(2-digit) Month(numeric) DayOfMonth Hour Minute Second”
-y	–doy=TIME	”Year DayOfYear SecondsOfDay”
-m	–mjd=TIME	”ModifiedJulianDate”
-o	–shortweekandsow=TIME	”10bitGPSweek SecondsOfWeek Year”
-z	–shortweekandzcounts=TIME	”10bitGPSweek ZCounts Year”
-f	–fullweekandsow=TIME	”FullGPSweek SecondsOfWeek”
-w	–fullweekandzcounts=TIMEo	”FullGPSweek ZCounts”
-u	–unixtime=TIME	”UnixSeconds UnixMicroseconds”
-Z	–fullZcounts=TIME	”fullZcounts”
-F	–format=ARG	Time format to use on output
-a	–add-offset=NUM	add NUM seconds to specified time
-s	–sub-offset=NUM	subtract NUM seconds from specified time

3.18.3 Examples

```
> timeconvert -r "05 06 1985 13:50:02"


      Month/Day/Year              5/6/1985
      Hour:Min:Sec                13:50:02
      Modified Julian Date        46191.576412037
      GPSweek DayOfWeek SecOfWeek  278 1 136202.000000
      FullGPSweek Zcount          278 90801
      Year DayOfYear SecondOfDay   1985 126 49802.000000
      Unix_sec Unix_usec          484235402 0
      FullZcount                  145842865
```

```
> timeconvert -o "1379 500 2006"


      Month/Day/Year              6/11/2006
      Hour:Min:Sec                00:08:20
      Modified Julian Date        53897.005787037
      GPSweek DayOfWeek SecOfWeek  355 0 500.000000
```

```
      FullGPSweek Zcount           1379 333
      Year DayOfYear SecondOfDay   2006 162 500.000000
      Unix_sec Unix_usec           1149984500 0
      FullZcount                   186122573
```

```
> timeconvert -o "1379 500 2006 -a 86400"


      Month/Day/Year               6/11/2006
      Hour:Min:Sec                 00:08:20
      Modified Julian Date         53897.005787037
      GPSweek DayOfWeek SecOfWeek  355 0 500.000000
      FullGPSweek Zcount           1379 333
      Year DayOfYear SecondOfDay   2006 162 500.000000
      Unix_sec Unix_usec           1149984500 0
      FullZcount                   186122573
```

```
> timeconvert -w "1381 500" -s 200


      Month/Day/Year               6/25/2006
      Hour:Min:Sec                 00:09:10
      Modified Julian Date         53911.0063657407
      GPSweek DayOfWeek SecOfWeek  357 0 550.000000
      FullGPSweek Zcount           1381 366
      Year DayOfYear SecondOfDay   2006 176 550.000000
      Unix_sec Unix_usec           1151194150 0
      FullZcount                   187171182
```

3.18.4 Notes

3.19 *WhereSat*

3.19.1 Overview

This application uses input ephemeris to compute the predicted location of a satellite. The Earth-centered, Earth-fixed (ECEF) position of the satellite is reported. Optionally, the topocentric coordinates–azimuth, elevation, and range–can be generated. The user can specify the time interval between successive predictions. Also the output can generated in a format easily imported into numerical packages.

3.19.2 Usage

Required Arguments

Short Arg.	Long Arg.	Description
-b	–broadcast=ARG	Specify a RINEX navigation file. The user may enter multiple files.
-p	–prn=NUM	Specify which SV to analuze.

Optional Arguments

Short Arg.	Long Arg.	Description
-h	–help	Generates help and usage.
-u	–position=ARG	Specify antenna position in ECEF (x,y,z) coordinates as "X Y Z". Used to give user-centered data (SV range, azimuth & elevation).
-s	–start=ARG	Specify time to begin analysis as "MO/DD/YYYY HH:MM:SS". The default is the end of the file.
-e	–end=ARG	Specify time to end analysis as "MO/DD/YYYY HH:MM:SS". The default is the beggning of the file.
-o	–output-filename=ARG	Outputs results to a MATLAB readable file.
-t	–time=NUM	Specify time increment for ephemeris calculation in seconds. Default is 900 (15 min.)

3.19.3 Examples

```
> WhereSat -b aira1720.06n -p 2 -u "918129.01 -4346070.45 803.18"
 -s "06/21/2006 17:00:00" -e "06/21/2006 20:00:00" -t 1800

Antenna Position: 918129 -4.34607e+06 803.18
Navigation File: aira1720.06n
Start Time:      06/21/2006 17:00:00
End Time:        06/21/2006 20:00:00
PRN:             2

Prn 2 Earth-fixed position and clock information:

Date      Time(UTC)  X (meters)        Y (meters)        Z (meters)
=============================================================================
06/21/2006 18:00:00 12758891.971859   18901201.616227    -14049016.596144
06/21/2006 18:30:00 12847888.097031   21541501.416411    -9315422.851798
```

```
 06/21/2006 19:00:00 12843576.989405   23087218.618683    -3957280.515764
 06/21/2006 19:30:00 12450313.769289   23516935.034029    1667186.089065

  . . .

   Clock Correc (s)
==================
    0.000007
    0.000007
    0.000007
    0.000007



 Data for user reference frame:

 Date      Time(UTC)  Azimuth      Elevation    Range to SV (m)
 ====================================================================
 06/21/2006 18:00:00 130.596202   -43.242769    29627531.177821
 06/21/2006 18:30:00 118.680085   -49.681012    29983796.522429
 06/21/2006 19:00:00 102.845663   -53.888528    30169796.433699
 06/21/2006 19:30:00 84.400419    -55.459042    30197072.648367

 Calculated 4 increments for prn 2 .
```

3.19.4 Notes

Index

calgps
application writeup, 22

fic2rin
application writeup, 25
ficafic
application writeup, 24
ficfica
application writeup, 24

mdp2fic
application writeup, 26
mdp2rinex
application writeup, 26
mergeRinFic
application writeup, 27
mergeRinMet
application writeup, 27
mergeRinNav
application writeup, 27
mergeRinObs
application writeup, 27

navdmp
application writeup, 28

poscvt
application writeup, 30
theory, 15
PRSolve
application writeup, 31

reszilla
application writeup, 33
RinexDump
application writeup, 35
rinexpvt
application writeup, 36
rtAshtech
application writeup, 39

timeconvert
application writeup, 40
theory, 13, 14

WhereSat
application writeup, 42