GPS Transformation Library Notes

Charles Rino

September 2010

1 Background

The problem of interest here is the real-world representation and manipulation of static and dynamic position data. Positions on the earth are determined by geodetic latitude, λ, longitude, φ, and height, h (llh). The datum against which geodetic coordinates are reported is necessarily some agreed upon surface surrogate. With the advent of GPS, the WGS84 ellipsoid has become the most commonly used reference. The specification of λ and φ with respect to a reference ellipsoidal surface is a straightforward extension of the spherical coordinates, but the math is more combersome. However, computatons involving directions and distances are much simpler in a rectangular coordinate system. The topocentric coordinate system (tcs) is a local cartesian system with the z-axis vertical, the x-axis positive along the eastward direction, and the y-axis positive along the northward direction. TCS is uniquely defined by the latitude, longitude, and height of the origin. Sensor data-space coordinates are unique to the particular instrumentation, its location, and its orientation. Sensor measurements are invariably translated (mapped) to a topocentric coordinate system, but to do so one must keep track of the surface location and the fact that each tcs system is unique to its own location. The collection of MATLAB utilities in the directory `GPS_Coordinate_Xforms` has been structured to perform such coordinate transformations and other related manipulations as described below.

The utilities are evolved shareware with multiple contributors. The book *Linear Algebra, Geodesy, and GPS* by Strang and Borre contains a list of functionally similar MATLAB utilities that can be downloaded from

http://www.i4.auc.dk/borre/matlab.

Commercial products such as the MATLAB mapping toolbox are available that will perform most of the manipulations with integrated graphics and GUIs. However, a set of vectorized GPS translation utilities with common formats is a useful supplement.

1.1 Geoid and Terrain Height

The reference ellipsoid is a reference surface against which terrain height can be reported directly. The geoid is a surface of constant gravitational potential chosen to coincide with mean sea level. Thus, a plum line at sea level will align itself normal to the geoid. The reference geoid is available on the internet as tables of latitude, longitude, and height with respect to the WGS84 ellipsoid. Commercially available terrain height maps are often reported as height data in universal transverse mercator (UTM) coordinates, which are conviently surface cartesian, but unique to a sector. For most post-survey applications, GPS and UTM coordinate conversion suffice. Thus, geoid manipulations are not included in this library. The utilities `utm2deg` and `wgs2utm` will convert from UTM to LLH and LLH to UTM coordinates, respectively.

1.2 Earth Centered Inertial Coordinates and Time

The information an Earth-based observer requires is where and when to look for an object in space. This requires both time and reference coordinate translation. Astronomers have confronted this problem for decades and adopted sidereal time, which is constructed to report the diurnal recurrence of an object fixed in space. It accounts for both the translational motion of the Earth over a one-day period and higher-order corrections as the applications demand. Consequently, sidereal time and geometric translations from the ecliptic plane must be used to convert the position and time in the ECI system to Earth-centered fixed coordinates (ECF) and universal time (UT); notationally $\mathbf{X}_{ECF}(UT)$ and $\mathbf{V}_{ECF}(UT)$.

2 The GPS Transformation Library

This section describes the structure of the main transformation utilities. An attempt has been made to make the utilities self explanatory.

2.1 Nomenclature

Geodedic coordinates (llh) are specified as $3 \times N$ arrays of column vectors

$$\begin{array}{ccc} \lambda_1 & & \lambda_N \\ \varphi_1 & \cdots & \varphi_N \\ h_1 & & h_N \end{array}$$

with latitude in radians (positive east), longitude in radians (positive north), and height with respect to the reference ellipsoid in meters. Topocentric coordinates (tcs) are specified as $3 \times N$ arrays of column vectors

$$\begin{array}{ccc} x_1 & & x_N \\ y_1 & \cdots & y_N \\ z_1 & & z_N \end{array}$$

where x is positive in the eastward direction at the topocentric origin, y is positive in the northward direction, and z is positive upward. Conversion of geodetic coordinates to topocentric coordinates and back requires specification of a geocentric source vector specifying latitude, longitude, and reference height, which may be a single vector for sequential conversion in a fixed system or a $3 \times N$ arrays of column vectors to accommodate a moving platform reference.

The xy plane is tangent to the WGS84 ellipsoid extrapolated to the reference height at the origin. For computations involving widely spaced points in space it is more convenient to use an earth-fixed coordinate system with its origin at the center of the earth (which coincides with the center of the refence ellipsoid). Earth centered fixed (ECF) coordinates are specified as $3 \times N$ arrays of column vectors

$$\begin{array}{ccc} X_1 & & X_N \\ Y_1 & \cdots & Y_N \\ Z_1 & & Z_N \end{array}$$

where X is fixed in the plane of the prime meridian (zero longitude), Y is perpendicular to X in the equatorial plane (zero latitude), and Z coincides with the earth's rotation axis.

2.2 Coordinate Transformations

Geodetic to ECF coordinates and the inverse transformation require no auxilary information. The vectorized routines `llh2ecfT_mks` and `ecf2llhT_mks` accept $3 \times N$ matrices of column vectors in the llh or ecf coordinates and return $3 \times N$ matrices of vectors in the complementary coordinate systems. A topocentric coordinate system is defined by its origin, which is specified as a geocentric vector. The routines `llh2tcsT_mks`, `ecf2tcsT`, and *`tcs2ecfT`* require a second argument, but otherwise work the same way as the ECF$\Longleftrightarrow$LLH transformation reoutines. The routines with `T` in their names have been vectorized to accommodate sequential transformations.

2.3 Rotations

A rotation is a unitary transformation of a rectangular coordinate system. Any such rotation can be initiated by individual rotations about the reference axes of the coordinate system to be transformed. The routine the z-axis, *Get_DirCos_Forward(A,MatrixFlavor)*, where is an angle A and *MatrixFlavor*, which is 1 for 'ROLL_TYPE' rotations about the x-axis, 2 for 'YAW_TYPE' rotations about the z-axis, or 3 for 'PITCH_TYPE' rotations about the y-axis. A 3×3 matrix of direction cosines is returned. The routines *ecf2uvw*, *tcs2uvw*, *uvw2ecs*, *uvw2llh*, and *uvw2tcs* use *Get_DirCos_Forward* to generate rotations to and from *uvw* systems with the x-axis rotated to coinside with the longitude of an origin vector in gedodec coordinates.

2.4 Special Purpose Utilities

For radar applications contacts are generally reported with range and bearing data space coordinates. Since range and bearing defines an arc in a plane defined by the bearing angle, one must determine the unique ray that intercepts the reference ellipsoid. There are actually two solutions, which are the real solutions to a fourth-order polynomial. Fourth-order polynomials admit an analytic solution, but it is simetimes more convenient to use a search. The routine *[stheta,ctheta]= Source2Ellipsoid(range,bearing,source,height)* calculates the sine and cosine of the computation from array inputs of range and bearing reports. Conversion to tcs coordinates can be recovered as

$$v\_tcs = range. * [stheta\prime. * sin(bearing);\ stheta\prime. * cos(bearing);\ ctheta\prime] \quad (1)$$

A simpler alternative, which is accurate enough for most ground-based radars is to assume

$$[stheta, ctheta] = [1, 0].$$

A spherical-earth correction can also be used. Note, however, that since the radar measures range in the atmosphere a 4/3 earth correction should not be used for that conversion. The utility *[range,stheta,ctheta]=GHorizon(bearing,source,height)* will compute the radar range to the horizon defined by the surface tangent ray.

3 Examples

A set of self-explanatory examples have been provided. To run the examples add `\GPS_CoordinateXforms` to the MATLAB path and transfer the MATLAB active directory to `\GPS_Xforms_Examples`. No additional MATLAB utilities are required to run the examples.