DETAILS OF THE VEHICLE MOTION GENERATOR

The **Script Motion Generator** within Tapestry creates highly accurate jerk limited dynamic-truth-data for a wide array of Host Vehicles. The use of the word *Script* is suggestive of words one-after-the-other combining to create a sentence. Similarly, one primitive maneuver-after-maneuver combines to make a Vehicle Trajectory (aka Vehicle Profile). The Trajectory is composed of Time, Position, Velocity, Acceleration, Jerk, Attitude, Attitude Rates, and Attitude Acceleration. The Vehicle Trajectory is used to control internal models for:

- Line-of-Sight Pseudorange Measurement formation.
- Antenna Lever Arm
- Antenna Gain and Phase-Shift Patterns
- Dead Reckoning Sensors
- GPS-SV Antenna Beam Shape
- Effects from Atmosphere, Ionosphere, and Ray Path.
- Etc.

Tapestry supports three methods for controlling one or both of two simulated vehicles.

- Importing at 10Hz Vehicle Trajectory in one of several format. See "*Import Data Formats* [PDF]" in the manuals folder.

- Generating Vehicle Motion real-time using a Closed Loop Interface such as ScramNet. See one of the "*Closed Loop Interface* …[PDF]" guides in the manuals folder.

- Using the Vehicle Motion Profile Generator within Tapestry.

This section describes the Tapestry Vehicle Trajectory Generator. Numerically it is realized using a 9^{TH} order Richardson Extrapolation Differential Equation Package[1]. The resulting ECEF state data is of extreme precession and differentiable through linear and angular jerk.

Vehicle Motion as realized by Tapestry is separated into three classes:

- **Terrestrial**. These are surface or near earth surface vehicles such as man packs, automobiles, and aircraft.
- **Ballistic**. These represent missiles, spinning munitions from artillery.
- **Spaceborne**. These are near-earth orbital vehicles.

For Terrestrial applications, motion primitives are combined within a *Script* to generate a time sequenced dynamics profile. Before we describe the methodology we define the coordinate frames.

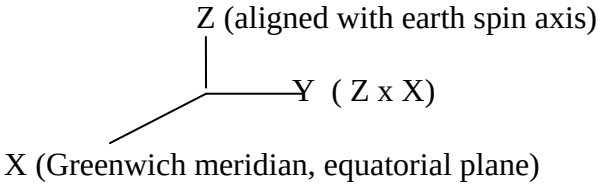
Coordinate Frames

[1] *Michael T. Heath, Scientific Computing, An Introductory Survey, 2nd edition, McGraw-Hill, New York, 2002*

For terrestrial applications, Tapestry has mechanized the following coordinate systems:

Earth-Centered-Earth-Fixed: (ECEF)

This coordinate system is used for the GPS satellite vectors and is the frame selected for the truth data output. It is defined, consistent with the WGS84 convention as follows:



Geographic (LLA):

This frame is used primarily to communicate with the user through the man machine interface. It uses latitude, longitude, and ellipsoidal altitude. It is a true North pointing frame.

Wander-Azimuth Locally Level Frame (Platform/NED)

The wander azimuth frame is the computational frame used for all terrestrial motion equations. This frame was selected to avoid the apparent singularity associated with a geographic (LLA) frame in which the vertical craft torque rate approaches infinity at the two poles.

The wander frame differs from the LLA frame in that North is not referenced to the earth spin axis (North Pole) except at the simulation start time. All terrestrial maneuvers are mechanized in the locally level wander azimuth frame with the wander angle offset computed when reference to a geographic frame is required. This frame generates great circle navigation when "cruising" at a constant wander heading. Note as the vehicle moves due (wander) east, the wander frame rotates clockwise relative to a true north pointing frame. The rotation rate is given by:

$$\omega = V_{EAST}/R\tan(\lambda) \qquad \text{(clockwise if Veast} > 0)$$

See Brockstein and Kuba[2] for the equations that describe the wander azimuth frame.

2 *Applied Mathematical Methods in Integrated Navigation Systems by Robert M. Rogers*

Script Motion Implementation

The Tapestry Script Motion Profile Generator implemented within Build Scenario via the Script Motion Keypad, uses the following design philosophy whenever possible:

(a) All maneuvers are built up from an analytic step-jerk representation in both the linear and angular regimes. This feature will be important in high dynamic applications in which code and carrier loop bandwidths/orders are selected based upon specific dynamic assumptions. The alternative use of step acceleration would severely stress not only a second but third order PLL or AFC loops. Step jerk provides a smooth and differentiable acceleration input into the receiver tracking loops.

The step jerk is specified in the wander azimuth frame and can be analytically integrated into acceleration and velocity. As described below, position (obtained from a direction cosine representation) - is not (in general) analytic in an ellipsoidal earth and is obtained by a numerical technique.

Angular motion is computed independent from the linear motion and is also based upon a step angular jerk. This allows the Tapestry user to precisely control the dynamics injected into the receiver under test resulting from motion of the antenna lever arm.

(b) All dynamic parameters (jerk, *acceleration, and velocity*) associated with the maneuvers are specified analytically in the local tangent wander frame with the exception of vehicle position. To compute position, an auto-step sixth order Romberg extrapolation/integration technique was used. This technique, developed by Bolrich and Stoer, results in an algorithm error of less than 0.01 PPM in those cases which can be checked analytically.

The use of analytic representations for the maneuvers allows the Tapestry Vehicle Motion Generator to achieve precise control of not only the linear dynamic state of the vehicle but the angular state as well. In the cases of extreme dynamics (in excess of 5 G's) the auto-step feature allows the Tapestry to partition the maneuver into multiple segments without any sacrifice in accuracy. In such high G situations, the Romberg integrator typically executes 36 function evaluations whereas a fourth order Runge-Kutta would perform only 4 evaluations. This increased operation count, while sacrificing some throughput, provides for an extreme amount of fidelity in the truth data.

(c) In order to synchronize the Tapestry RF outputs with the truth data a dynamic parameter readjustment is often performed. This synchronization is required because the Tapestry generates RF signals derived from updates to a numeric-controlled-oscillator (NCO) at 500 Hz based on interpolation of 10 Hz polynomial parameters. In addition, the user entered vehicle maneuvers are synchronized such that they always start and stop on an NCO reset epoch. This precise synchronization assures the Tapestry user that no maneuver will persist (in the RF channel) past the time at which the maneuver was supposed to terminate. This is extremely important in

situations where Tapestry provides test data which is to be synchronized with the RF signals. The synchronization of maneuver start and stop times to 10 Hz boundaries often requires Tapestry to change the parameters (typically the applied jerk) associated with a maneuver. These changes in the values of dynamic parameters are required such that the terminal conditions desired by the user can be achieved.

> *Note: Vehicle parameter readjustment is in a direction that always <u>decreases</u> the dynamic stress rather than increases it. The operator can be assured that the specified vehicle "maximum" parameters are never exceeded during a maneuver.*

All maneuvers are passed through a pre-processing function in which:

> (a). The user parameter inputs are validated against the vehicle dynamic constraints.
> (b). The maneuver is decomposed into its constituent jerk primitives. A jerk primitive is defined as a time-slice in which the applied jerk is constant. The constant value can be either positive, negative or zero.
> (c). The time intervals for each jerk primitive are computed - consistent with the alignment of NCO and maneuver epochs discussed previously. Jerk magnitude may be re-adjusted if required.

Given a decomposed jerk primitive set, and times, the vehicle (wander-azimuth) navigation state is computed schematically as:

J_I = I-th jerk primitive (Wander-NED)
A_I = I-th Acceleration increment (Wander-NED)
V_I = I-th Velocity increment (Wander-NED)
C^E_P = Position direction cosines relating Platform to ECEF

The differential equations for the update of C^E_P is given by:

for(all jerk primitives) {

J_I = re-scaled jerk primitive (I)

A_I = $A_{(I-1)} + \int J_I\,(t)dt$

V_I = $V_{I-1} + \iint J_I\,(t)dt + \int A_{(I-1)}dt$

$\dot{(C^E_P)}$ = $C^E_P\,(\omega^P_{EP}\,x)$

where the skew symmetric matrix $(\omega^P_{EP}\,x) = \begin{vmatrix} 0 & -\omega_D & \omega_E \\ & 0 & -\omega_N \\ & & 0 \end{vmatrix}$

and:

R_M = radius prime meridian (function of C^E_P)
R_V = radius prime vertical (function of C^E_P)
α = wander angle(function of C^E_P)

$$\omega^{N}_{EP} = V_E (\cos(\alpha)\cos(\alpha)/R_V + \sin(\alpha)\sin(\alpha)/R_M) + V_N(\cos(\alpha)\sin(\alpha) (1/R_V - 1/R_M)$$

$$\omega^{N}_{EP} = -V_N (\sin(\alpha)\sin(\alpha)/R_V + \cos(\alpha)\cos(\alpha)/R_M) - V_E(\cos(\alpha)\sin(\alpha) (1/R_V - 1/R_M)$$

$$\omega^{D}_{EP} = 0.0$$

} (Increment I for next primitive)

Unfortunately, these differential equations are coupled and non-linear. As mentioned previously, the solution to the position direction cosine update is obtained using a numerical integration routine coded from the IEEE proceeding by Bolrich and Stoer for solving simultaneous coupled non-linear equations. The accuracy of this technique is superb exceeding most 8-th order algorithms.

Overview of modeled maneuvers

The following presents a brief description of the analytics associated with the maneuver generation algorithm for Tapestry.

Cruise

The **Cruise** propagates the current vehicle navigation state (based upon the terminal conditions of the maneuver immediately preceding the selection of the **Cruise**) via great-circle navigation for a user specified time interval.

Change Speed

Change Speed causes the user vehicle to accelerate (or decelerate) from the terminal speed of the preceding maneuver to the speed selected by the operator. The speed is changed via a three segment process.

1) The jerk up phase: Applied constant jerk for a period sufficient to achieve the desired vehicle acceleration. After achieving the desired acceleration the jerk is instantaneously brought to zero.
2) The constant acceleration phase: This phase will have zero vehicle jerk and constant acceleration.
3) The jerk down phase: This phase will have an applied step jerk which will remove the constant vehicle acceleration. At the termination of this phase, the time-rate-of-change of velocity will be zero and the final vehicle speed will be precisely as specified by the user. As in the other phases, the position is propagated forward in time.

In a situation where the specified vehicle acceleration is very large, it may be possible to achieve the desired terminal velocity without achieving the desired terminal acceleration. In this case the second phase is not executed.

Accelerate

Accelerate results in a step jerk applied for an interval in order to reach the desired acceleration. During the jerk phase (positive/negative for a positive/negative acceleration change) the vehicle velocity changes quadratically and the position direction cosines are propagated numerically.

Turn

Turn allows the user to change the heading of the simulated vehicle while precisely controlling the radial jerk, acceleration, and the presence of banking. The ground track of the **Turn** maneuver closely approximates a circular arc overlaid on the earth at some fixed altitude. The deviation from an exact circular arc is caused by the desired radial acceleration being achieved through the use of a ramp rather than a step.

Nominally the **Turn** is executed in three linear motion and seven angular motion phases. The linear and angular motions are performed simultaneously with the linear motion sequenced as follows:

1) The jerk to constant radial acceleration phase: This phase applies a constant radial jerk perpendicular to the instantaneous vehicle velocity. The jerk persists for an interval sufficient to achieve the acceleration specified by the user.
2) The constant radial acceleration phase: During this phase the vehicle executes a circular motion with the constant radial acceleration achieved during the previous jerk segment. The applied radial jerk during this phase is zero, however the tangential jerk is not zero (a certain level of jerk is required to swing the acceleration vector throughout its circular motion).
3) The jerk to zero radial acceleration phase: This phase is the inverse of the first phase and is required to remove the applied radial acceleration.

Turn causes the vehicle bank angle (roll) to change (if specified). The roll angle is computed automatically by setting the terminal value to:

$U = \tan^{-1}(A/G)$, where A is the applied radial acceleration and G is the acceleration due to gravity.

The vehicle is rolled through this bank angle in seven phases as follows:

1) The jerk to constant angular rate phase: During this phase the vehicle attitude is computed by dividing the interval (quantized) into two segments.

During the first segment a positive (clockwise) angular jerk is applied (assuming a right-turn) and during the second phase a counter-clockwise angular jerk is applied. These two segments cause the vehicle to achieve a constant roll rate about the nose.
2) The constant angular rate phase: The vehicle continues to roll at a constant rate without applied acceleration or jerk.
3) The jerk to zero rate phase: This phase reverses the sequencing of phase (1) in order to remove the applied vehicle roll rate. At the termination of this phase the vehicle is banked at the desired bank-angle.
4) Constant bank phase: During this phase the vehicle is held at a constant bank angle. The angular rate, acceleration, and jerk are zero.
5), 6), 7). These phases are the inverse of, respectively, phases 3, 2, and 1.

At the termination of the seven attitude phases the vehicle is back in a level configuration.

The **Turn** sequencing described above applies during the nominal **Turn** scenario. Deviations can result if the turn angle is very shallow or the applied acceleration is very large. In such situations the first process to occur is that the constant radial acceleration phase is removed and the applied jerk readjusted - if this suffices the maneuver is executed, otherwise an error window is presented. Similar conditions can occur such that the bank angle must be achieved without a constant angular rate phase.

Pitch

Pitch mechanization and sequencing is very similar to the **Turn** maneuver. This similarity is due to the **Pitch** *being* a **Turn** maneuver in the vertical (rather than the horizontal) plane. The **Pitch** can either be "up" or "down" depending upon the sense of the pitch angle entered by the user.

Climb

The **Climb** maneuver is a compound maneuver constructed by serially sequencing a **Pitch**, a **Cruise** and another **Pitch**. During this the **Climb** is executed such that the vehicle maintains a constant speed.

The **Climb**, however, is more complicated than just splicing these three maneuvers together. The complication results from the terminal height at the end of the two **Pitch** phases not being given analytically (it actually is an ERF function which is only computed in tabular form). This lack of an analytical solution results in an iterative process being used, in which Tapestry tries to determine the duration of the **Pitch** phases based upon a numerical integration of a transcendental function. This process causes the **Climb** to take very slightly more throughput than other maneuvers. At the completion of the iterative process the precise times are obtained for the two **Pitch**

phases with the duration of the altitude change "tuned" by the **Cruise** phase (executed during a constant pitch angle which accumulates both an altitude as well as a down track change). The time associated with the cruise phase is, however, in general not quantized to the required 10 Hz time epochs. The subsequent quantization requires a readjustment of the duration of the **Pitch** phases (followed by a readjustment of the **Cruise** phase). This process is, in general, virtually never ending and the Tapestry compromises by executing the **Climb** in such a way so as to achieve a terminal altitude as *close as possible* to that specified by the user. Typically this error is within a few meters.

It is important to note however, that even though the terminal height achieved during the **Climb** is not *exactly* as specified by the user, it is, however, executed exactly! This means that whatever terminal condition was achieved, it was achieved precisely and all truth and GPS data are consistent. What this means is that the dynamics and pseudorange associated with the climb are precisely in agreement with the climb as executed by Tapestry.

Problems with the **Climb** can result when either a very small altitude change is desired (100-200 meters) while pulling large G's, or when a steep pitch angle is specified with insufficient G's being specified.

The specification of a large pitch angle can cause the **Climb** to exceed the terminal height before the pitch can be completed. In general the software will do the best it can.

Roll

The **Roll** maneuver is essentially one half of the bank maneuver described in the **Turn** maneuver.

The **Roll** maneuver leaves the vehicle non level. Other maneuvers following the **Roll** will maintain the roll achieved during this maneuver. In some cases a non-zero initial roll will cause some maneuvers to be unable to satisfy the user specified terminal conditions.

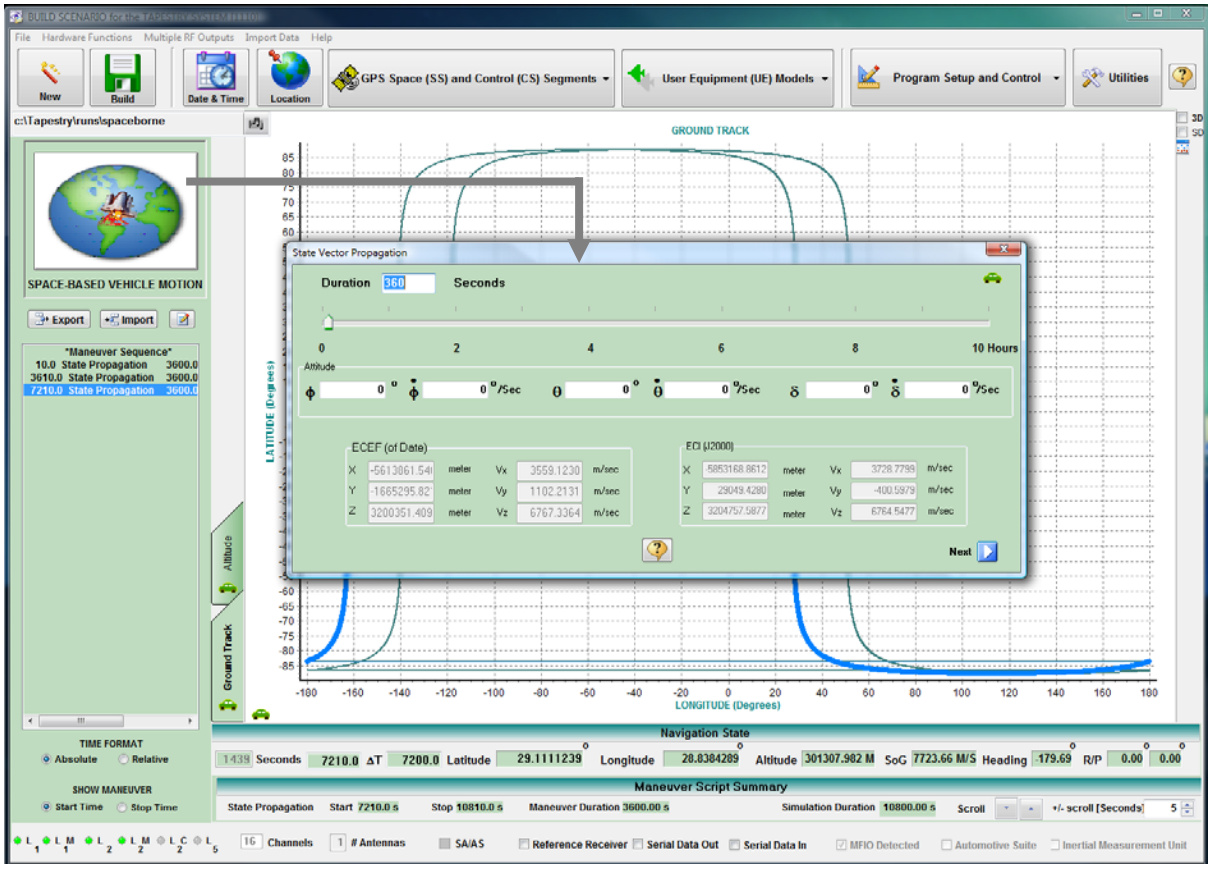
The Ballistic Projectile Model

The **Ballistic-Projectile** maneuver facilitates the testing of spinning GPS guided munitions. The figure below illustrates the parameters associated with the model.

Refer to this document for a detailed description of this maneuver:

..\..\..\..\Tapestry\Documentation\Manuals\BallisticModel.pdf

(Spaceborne) State Vector Propagation



The following coupled differential equation is solved using the Richardson Extrapolator discussed previously.

State Vector:

$$\partial \vec{P}^{ECI}/\partial t = \vec{V}^{ECI};$$
$$\partial \vec{V}^{ECI}/\partial t = \vec{A}^{ECI};$$

Thrust Vector in ORBIT Frame (DXR)

$$\text{With } \vec{A}^{ECI} = \vec{G}^{ECI} + C^{I}_{O}\vec{T}^{O} - \alpha \vec{V}^{ECI}\bullet\vec{V}^{ECI}$$

Gravity

Drag

Attitude:

$$\dot{(C^{B}_{I})} = C^{B}_{I}(\omega^{B}_{IB}\,x)$$

ORBIT Locally Level Frame to ECI direction cosines

$$(\omega^{B}_{IB}\,x) = \begin{vmatrix} 0 & -\omega^{X}_{IB} & \omega^{Y}_{IB} \\ & 0 & -\omega^{Z}_{IB} \\ & & 0 \end{vmatrix}$$

ECI to BODY direction cosines (*Attitude*)