3D Rotation for Flight Simulator - Google Earth Application

R.P. McElrath, October, 2015

Camera rotation for the Flight Simulator - Google Earth (G.E.) swivel cam application can be computed using either Quaternion or Euler rotation algorithms.

Input data are the same for either method: 1) Euler angle triplet from Flight Simulator (`A:PLANE HEADING DEGREES TRUE, A:ATTITUDE INDICATOR PITCH DEGREES, A:ATTITUDE INDICATOR BANK DEGREES`) plus 2) Euler triplet of the camera swivel (Yaw, Tilt, Roll) via input from the HAT or POV switch on a joystick that returns camera view angles relative to the current aircraft axes (e.g., “look left”).

The required output is the Heading-Tilt-Roll Euler angle triplet needed to aim the Google Earth camera relative to the fixed world reference frame that G.E. uses.

The rotation process steps are:

1) Convert the input Euler angle triplets to either two Quaternions or two Euler Rotation Matrices

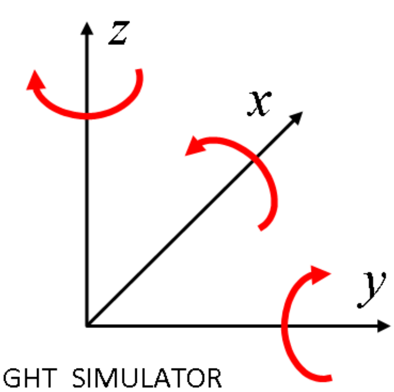
2) Calculate the compound (or concatenated or stacked) rotation through a rotation multiplication process (Quaternion Hamilton multiplication or Euler Rotation Matrix multiplication)

3) Convert the resulting multiplication back to Euler Heading, Tilt, and Roll angles

This paper assumes Euler angles are in Tait-Bryan Yaw, Pitch, and Roll form. The rotations are intrinsic and active (alibi) and the rotation order is z then y then x.

Equations presented here fit the coordinate system conventions of Flight Simulator (left hand Cartesian coordinate system and rotations) and Google Earth (mixed coordinate system). The equations will differ from those published in other sources if different conventions are used.

FLIGHT SIMULATOR

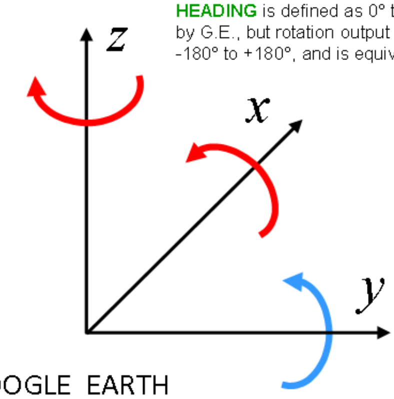


FLIGHT SIMULATOR
LEFT HAND COORDINATE SYSTEM

	AXIS	POSITIVE	RANGE
HEADING	z	C.W.	0° to 360°
PITCH	y	DOWN	-90° to +90°
BANK	x	LEFT	-180° to +180°

GOOGLE EARTH

HEADING is defined as 0° to 360° by G.E., but rotation output is -180° to +180°, and is equivalent



GOOGLE EARTH
MIXED COORDINATE SYSTEM

	AXIS	POSITIVE	RANGE
HEADING	z	C.W.	**-180° to +180°**
TILT	y	**UP**	**0° to 180°**
ROLL	x	LEFT	-180° to +180°

Quaternion Rotation

Convert Euler (Tait-Bryan) Angles to a Quaternion

$$q_0 = \cos\left(\frac{w}{2}\right)\cos\left(\frac{v}{2}\right)\cos\left(\frac{u}{2}\right) + \sin\left(\frac{w}{2}\right)\sin\left(\frac{v}{2}\right)\sin\left(\frac{u}{2}\right) \qquad (1a)$$

$$q_1 = \sin\left(\frac{w}{2}\right)\cos\left(\frac{v}{2}\right)\cos\left(\frac{u}{2}\right) - \cos\left(\frac{w}{2}\right)\sin\left(\frac{v}{2}\right)\sin\left(\frac{u}{2}\right) \qquad (1b)$$

$$q_2 = \cos\left(\frac{w}{2}\right)\sin\left(\frac{v}{2}\right)\cos\left(\frac{u}{2}\right) + \sin\left(\frac{w}{2}\right)\cos\left(\frac{v}{2}\right)\sin\left(\frac{u}{2}\right) \qquad (1c)$$

$$q_3 = \cos\left(\frac{w}{2}\right)\cos\left(\frac{v}{2}\right)\sin\left(\frac{u}{2}\right) - \sin\left(\frac{w}{2}\right)\sin\left(\frac{v}{2}\right)\cos\left(\frac{u}{2}\right) \qquad (1d)$$

where:

u = roll angle.	Flight Simulator u = Bank	Swivel Camera u = Roll
v = pitch angle.	Flight Simulator v = Pitch	Swivel Camera v = Tilt
w = yaw angle.	Flight Simulator w = Heading	Swivel Camera w = Yaw

Compound Rotation: Quaternion Multiplication

The compound (concatenated) rotation, Flight Simulator Heading-Pitch-Bank rotation followed by Camera Yaw-Tilt-Roll swivel, is also represented by a quaternion. The compound rotation quaternion (***t***) is calculated by quaternion multiplication:

t = Flight Simulator H-P-B quaternion (***r***) * Camera Y-T-R quaternion (***s***)

$$(t_0, t_1, t_2, t_3) = (r_0, r_1, r_2, r_3) * (s_0, s_1, s_2, s_3)$$

where ***t*** is the Hamilton product whose coefficients are:

$$t_0 = (r_0 s_0 - r_1 s_1 - r_2 s_2 - r_3 s_3) \qquad (2a)$$

$$t_1 = (r_0 s_1 + r_1 s_0 - r_2 s_3 + r_3 s_2) \qquad (2b)$$

$$t_2 = (r_0 s_2 + r_1 s_3 + r_2 s_0 - r_3 s_1) \qquad (2c)$$

$$t_3 = (r_0 s_3 - r_1 s_2 + r_2 s_1 + r_3 s_0) \qquad (2d)$$

Quaternion coefficients r_0, r_1, r_2, r_3 and s_0, s_1, s_2, s_3 are calculated using equations 1 a-d. For Flight Simulator, $r_n = q_n$. For the swivel camera, $s_n = q_n$.

Convert Quaternion to Euler (Tait-Bryan) Angles

TILT: $-1*\text{asin}[\ 2*(\ t_0*t_2 - t_1*t_3\ )\ ]$ (3)

ROLL: $\text{atan2}[\ t_0^2 + t_1^2 - t_2^2 - t_3^2\ ,\ 2*(\ t_0*t_3 + t_1*t_2\ )\ ]$ **(4)**

HDG : $\text{atan2}[\ t_0^2 - t_1^2 - t_2^2 + t_3^2\ ,\ 2*(\ t_0*t_1 + t_2*t_3\ )\ ]$ **(5)**

To prevent #NUM! Out of Range errors if using Excel, TILT should include a ROUND function, e.g., $-1*\text{asin}[\ 2*(\ \text{ROUND}(\ t_0*t_2 - t_1*t_3, 13\ )\ )\ ]$ *with num_digits greater than zero. 13 is a good choice.*

Mathematic Singularity ("Gimbal Lock")

Quaternions are immune from the phenomenon of "gimbal lock"; however, Euler angles are not. The issue is a mathematical singularity occurring when TILT equals ± 90.0° and the ROLL and HDG equations both equal atan2(0, 0) and are therefore invalid.

To mitigate the singularity at ± 90° TILT:

TILT *(as derived from* ***t****)*: $-1*\text{asin}[\ 2*(\ t_0*t_2 - t_1*t_3\ )\ ]$ *(no change)*

ROLL *(at TILT = ±90°)*: $\equiv 0°$

HDG *(at TILT = +90°)*: $= -2*\text{atan2}(\ t_1, t_0\ )$
HDG *(at TILT = −90°)*: $= 2*\text{atan2}(\ t_0, t_1\ )$

The singularity can either be ignored*, or equations (4) and (5) modified to conditionally account for it. Actually, it would be quite rare for equation (3) to yield exactly ±90° in the first place, and if it does, then on the next Flight Simulator gauge update cycle (~56 ms later), it's very likely to no longer be ±90° and everything is back to normal.

Google Earth Tilt

Lastly, the Google Earth Tilt range convention is addressed by adding 90° to results of equation (3):

GE Tilt = TILT + 90° **(6)**

Unlike Flight Simulator Pitch, Google Earth Tilt ranges from 0° to +180°, with no negative values. In Google Earth, a Tilt of 0° looks down, 90° is in the x-y plane, and +180° is in the 'sky'. In level flight, Flight Simulator returns a Pitch of near 0°, but to Google Earth, 0° Tilt points the camera down. Consequently, to achieve the pilot's perspective of looking forward rather than down, 90° should be added to TILT from equation (3).

Equations **4**, **5**, and **6** are used for the Google Earth view as Flight Simulator is running. The full set of equations needed is **1a-d** for the Flight Sim quaternion, **1a-d** for the Camera quaternion, **2a-d** for the Flight Sim-Camera Hamilton product, and **4**, **5**, and **6** for the final Google Earth Hdg, Tilt, and Roll angles.

Euler Rotation

Convert Euler (Tait-Bryan) Angles to a Rotation Matrix

The Elemental rotation matrices are:

$$\boldsymbol{R}_x(u) = \begin{vmatrix} 1 & 0 & 0 \\ 0 & \mathrm{c}(u) & -\mathrm{s}(u) \\ 0 & \mathrm{s}(u) & \mathrm{c}(u) \end{vmatrix}$$

$$\boldsymbol{R}_y(v) = \begin{vmatrix} \mathrm{c}(v) & 0 & \mathrm{s}(v) \\ 0 & 1 & 0 \\ -\mathrm{s}(v) & 0 & \mathrm{c}(v) \end{vmatrix}$$

$$\boldsymbol{R}_z(w) = \begin{vmatrix} \mathrm{c}(w) & -\mathrm{s}(w) & 0 \\ \mathrm{s}(w) & \mathrm{c}(w) & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

where:

u = roll angle	Flight Simulator u = Bank	Swivel Camera u = Roll
v = pitch angle	Flight Simulator v = Pitch	Swivel Camera v = Tilt
w = yaw angle	Flight Simulator w = Hdg	Swivel Camera w = Yaw

c( ) = cosine function and s( ) = sine function

The resulting Rotation Matrix is the product of the Elemental matrices:

$\boldsymbol{R}_z(w) * \boldsymbol{R}_y(v) * \boldsymbol{R}_x(u)$

All are 3 X 3 matrices, so the matrix multiplication produces a 3 X 3 matrix:

$$\boldsymbol{R}_z(w)\boldsymbol{R}_y(v)\boldsymbol{R}_x(u) = \begin{vmatrix} \mathrm{c}(v)\mathrm{c}(w) & \mathrm{s}(u)\mathrm{s}(v)\mathrm{c}(w) - \mathrm{c}(u)\mathrm{s}(w) & \mathrm{s}(u)\mathrm{s}(w) + \mathrm{c}(u)\mathrm{s}(v)\mathrm{c}(w) \\ \mathrm{c}(v)\mathrm{s}(w) & \mathrm{c}(u)\mathrm{c}(w) + \mathrm{s}(u)\mathrm{s}(v)\mathrm{s}(w) & \mathrm{c}(u)\mathrm{s}(v)\mathrm{s}(w) - \mathrm{s}(u)\mathrm{c}(w) \\ -\mathrm{s}(v) & \mathrm{s}(u)\mathrm{c}(v) & \mathrm{c}(u)\mathrm{c}(v) \end{vmatrix}$$

Matrix multiplication order is important: $\boldsymbol{R}_z(w) * \boldsymbol{R}_y(v) * \boldsymbol{R}_x(u)$. In Excel, it would be written

=MMULT($\boldsymbol{R}_z(w)$,MMULT($\boldsymbol{R}_y(v)$, $\boldsymbol{R}_x(u)$))

where $\boldsymbol{R}_z(w)$, $\boldsymbol{R}_y(v)$, and $\boldsymbol{R}_x(u)$ are replaced by the 3 row by 3 column range definitions (e.g., A1:C3) for each Elemental matrix.

Compound Rotation: Rotation Matrix Multiplication

The objective Google Earth camera orientation is a combination of two sequential rotation sets:

1) Flight Simulator aircraft attitude followed by

2) Camera swivel relative to the aircraft axes

The Rotation Matrices involved are:

Flight Sim Attitude: $\boldsymbol{R}_{FS} = \boldsymbol{R}_z(\text{Hdg}) * \boldsymbol{R}_y(\text{Pitch}) * \boldsymbol{R}_x(\text{Bank})$

Camera Swivel: $\boldsymbol{R}_{CAM} = \boldsymbol{R}_z(\text{Yaw}) * \boldsymbol{R}_y(\text{Tilt}) * \boldsymbol{R}_x(\text{Roll})$

The compound, or concatenated, rotation is calculated by rotation matrix multiplication:

$$\boldsymbol{R}_{FS\times CAM} = \boldsymbol{R}_{FS} * \boldsymbol{R}_{CAM} \tag{10}$$

or, equivalently, expressed as the product of the individual Elemental rotation matrices:

$$\boldsymbol{R}_{FS\times CAM} = \boldsymbol{R}_z(\text{Hdg}) * \boldsymbol{R}_y(\text{Pitch}) * \boldsymbol{R}_x(\text{Bank}) * \boldsymbol{R}_z(\text{Yaw}) * \boldsymbol{R}_y(\text{Tilt}) * \boldsymbol{R}_x(\text{Roll})$$

This also results in a 3 X 3 matrix. The matrix elements are labeled as

$$\boldsymbol{R}_{FS\times CAM} = \begin{vmatrix} \boldsymbol{r}_{11} & \boldsymbol{r}_{12} & \boldsymbol{r}_{13} \\ \boldsymbol{r}_{21} & \boldsymbol{r}_{22} & \boldsymbol{r}_{23} \\ \boldsymbol{r}_{31} & \boldsymbol{r}_{32} & \boldsymbol{r}_{33} \end{vmatrix}$$

Matrix multiplication of equation (10) yields the following elements that are provided in the event the user's software doesn't include an adequate matrix or vector library.

NOTE: *In the equations below, Hdg, Pitch, Bank, and Yaw, Tilt, Roll are the angles used in the Elemental rotation matrices, equations (7), (8), and (9). They should not be confused with the compound Google Earth angles of equations (11), (12), and (13).*

*For clarity, "negative sine" is shown as −1*sin( ).*

r_{11} = cos(Pitch) * cos(Hdg) * cos(Tilt) * cos(Yaw) + cos(Tilt) * sin(Yaw) * sin(Bank) * sin(Pitch) * cos(Hdg) − cos(Tilt) * sin(Yaw) * cos(Bank) * sin(Hdg) − sin(Tilt) * sin(Bank) * sin(Hdg) − sin(Tilt) * cos(Bank) * sin(Pitch) * cos(Hdg)

r_{12} = cos(Pitch) * cos(Hdg) * sin(Roll) * sin(Tilt) * cos(Yaw) − cos(Pitch) * cos(Hdg) * cos(Roll) * sin(Yaw) + sin(Bank) * sin(Pitch) * cos(Hdg) * cos(Roll) * cos(Yaw) + sin(Bank) * sin(Pitch) * cos(Hdg) * sin(Roll) * sin(Tilt) * sin(Yaw) − cos(Bank) * sin(Hdg) * cos(Roll) * cos(Yaw) − cos(Bank) * sin(Hdg) * sin(Roll) * sin(Tilt) * sin(Yaw) + sin(Bank) * sin(Hdg) * sin(Roll) * cos(Tilt) + cos(Bank) * sin(Pitch) * cos(Hdg) * sin(Roll) * cos(Tilt)

r_{13} = cos(Pitch) * cos(Hdg) * sin(Roll) * sin(Yaw) + cos(Pitch) * cos(Hdg) * cos(Roll) * sin(Tilt) * cos(Yaw) + sin(Bank) * sin(Pitch) * cos(Hdg) * cos(Roll) * sin(Tilt) * sin(Yaw) − sin(Bank) * sin(Pitch) * cos(Hdg) * cos(Roll) * sin(Tilt) * sin(Yaw) − cos(Bank) * sin(Hdg) * cos(Roll) * sin(Tilt) * sin(Yaw) + cos(Bank) * sin(Hdg) * sin(Roll) * cos(Yaw) + sin(Bank) * sin(Hdg) * cos(Roll) * cos(Tilt) + cos(Bank) * sin(Pitch) * cos(Hdg) * cos(Roll) * cos(Tilt)

r_{21} = cos(Pitch) * sin(Hdg) * cos(Tilt) * cos(Yaw) + cos(Tilt) * sin(Yaw) * cos(Bank) * cos(Hdg) + cos(Tilt) * sin(Yaw) * sin(Bank) * sin(Pitch) * sin(Hdg) − sin(Tilt) * cos(Bank) * sin(Pitch) * sin(Hdg) + sin(Tilt) * sin(Bank) * cos(Hdg)

r_{31} = −1*sin(Pitch) * cos(Tilt) * cos(Yaw) + sin(Bank) * cos(Pitch) * cos(Tilt) * sin(Yaw) + cos(Bank) * cos(Pitch) * −1*sin(Tilt)

r_{32} = −1*sin(Pitch) * sin(Roll) * sin(Tilt) * cos(Yaw) − cos(Roll) * sin(Yaw) * −1*sin(Pitch) + sin(Bank) * cos(Pitch) * cos(Roll) * cos(Yaw) + sin(Bank) * cos(Pitch) * sin(Roll) * sin(Tilt) * sin(Yaw) + cos(Bank) * cos(Pitch) * sin(Roll) * cos(Tilt)

r_{33} = −1*sin(Pitch) * sin(Roll) * sin(Yaw) − sin(Pitch) * cos(Roll) * sin(Tilt) * cos(Yaw) + sin(Bank) * cos(Pitch) * cos(Roll) * sin(Tilt) * sin(Yaw) − sin(Bank) * cos(Pitch) * sin(Roll) * cos(Yaw) + cos(Bank) * cos(Pitch) * cos(Roll) * cos(Tilt)

Convert Rotation Matrix to Euler (Tait-Bryan) Angles

The compound rotation angles required to properly orient the Google Earth swivel camera while the aircraft is flying are derived from the $\boldsymbol{R}_{FS\times CAM}$ rotation matrix:

TILT:	$v = \sin^{-1}(\, r_{31}\,)$	$= \mathrm{asin}(\, r_{31}\,)$	(11)
ROLL:	$u = \tan^{-1}(\, r_{33} / r_{32}\,)$	$= \mathrm{atan2}(\, r_{33}, r_{32}\,)$	**(12)**
HDG:	$w = \tan^{-1}(\, r_{11} / r_{21}\,)$	$= \mathrm{atan2}(\, r_{11}, r_{21}\,)$	**(13)**

Mathematic Singularity ("Gimbal Lock")

Rotation Matrices are immune from the phenomenon of "gimbal lock"; however, Euler angles are not. The issue is a mathematical singularity occurring when $\boldsymbol{r}_{31} = \pm 1$ and $\boldsymbol{r}_{11} = \boldsymbol{r}_{21} = \boldsymbol{r}_{32} = \boldsymbol{r}_{33} = 0$ which yields a TILT value of ± 90.0°, but the ROLL and HDG equations both equal atan2(0, 0) and are therefore invalid.

To mitigate the singularity at ± 90° TILT:

TILT *(as derived from* $\boldsymbol{R}_{FS\times CAM}$*)*: $= \text{asin}(\ \boldsymbol{r}_{31}\ )$ *(no change)*

ROLL *(at TILT = ±90°)*: $\equiv 0°$

HDG *(at TILT = +90°)*: $= \text{atan2}(\ \boldsymbol{r}_{13}, \boldsymbol{r}_{12}\ )$
HDG *(at TILT = -90°)*: $= -1 * \text{atan2}(\ \boldsymbol{r}_{13}, \boldsymbol{r}_{12}\ )$

The singularity can either be ignored*, or equations (12) and (13) modified to conditionally account for it. Actually, it would be quite rare for equation (11) to yield exactly ±90° in the first place, and if it does, then on the next Flight Simulator gauge update cycle (~56 ms later), it's very likely to no longer be ±90° and everything is back to normal.

* In hundreds of aerobatic Loops recorded at the default 18 hz gauge update cycle rate, never has Flight Simulator returned ±90.00° for `A:ATTITUDE INDICATOR PITCH DEGREES`. Naturally in a Loop, the pitch A:Vars returned by Flight Simulator approach ±90°, within tenths or, at best and *very* infrequently, within hundredths of a degree, but never precisely equal ±90.00°.

On the other hand, ROLL and HDG Euler angles calculated from equations (4) and (5) or (12) and (13) are stable at TILT values extremely close (but not equal) to ±90°, without applying the singularity mitigation formulas. For example, TILT = 89.99999999999° or 90.00000000001° both yield stable and accurate values of ROLL and HDG from equations (4) and (5) or (12) and (13) without applying the singularity solutions.

The conclusion is that Flight Simulator may never return a Pitch value that causes a gimbal lock problem with the G.E. rotations, even considering the compound rotation operation. This is why it's probably acceptable to ignore the singularity issue in this application.

Google Earth Tilt

Lastly, the Google Earth Tilt range convention is addressed by adding 90° to results of equation (11):

$$\text{GE Tilt} = \text{TILT} + 90° \qquad \textbf{(14)}$$

Unlike Flight Simulator Pitch, Google Earth Tilt ranges from 0° to +180°, with no negative values. In Google Earth, a Tilt of 0° looks down, 90° is in the x-y plane, and +180° is in the 'sky'. In level flight, Flight Simulator returns a Pitch of near 0°, but to Google Earth, 0° Tilt points the camera down. Consequently, to achieve the pilot's perspective of looking forward rather than down, 90° should be added to TILT from equation (11).

Equations **12**, **13**, and **14** are used for the Google Earth view as Flight Simulator is running.

Six Degrees of Freedom

Flight Simulator and Google Earth camera views require definition of all six degrees of freedom (DOF): three rotational DOF plus three translational DOF. Rotations and compound rotations have been discussed up to this point.

The remaining DOF are translational: Longitude, Latitude, and Altitude. The translational values are simply the Flight Simulator variables `A:PLANE LATITUDE`, `A:PLANE LONGITUDE`, and `A:PLANE ALTITUDE`.

A typical Google Earth camera orientation would be written as follows:

```
<Camera>
  <longitude>-155.056566267712</longitude>
  <latitude>19.7143216472382</latitude>
  <altitude>1124.20057499758</altitude>
  <heading>10.5365345501778</heading>
  <tilt>100.656068754235</tilt>
  <roll>4.53358066926012</roll>
  <altitudeMode>absolute</altitudeMode>
</Camera>
```

where heading, tilt, and roll are the compound rotation Euler angles previously discussed.

The Google Earth kml schema assumes 'degrees' units for longitude, latitude, heading, tilt and roll and 'meters' units for altitude.

Absolute altitude mode means that the altitude value is measured relative to mean sea level, which is the same reference datum used by `A:PLANE ALTITUDE`.

Technical Note: atan2 Argument Order

Equations in this paper were evaluated using Microsoft Excel. Spreadsheets such as Excel reverse the order of arguments in the atan2 function compared to JavaScript. The conventional order, atan2(x, y) is written as atan2(y, x) in Excel and in this paper.

Consequently, to use the atan2 expressions found here in JavaScript and potentially other languages, please reverse the order of the arguments.

Quaternion Rotation Example
Compound Rotation

AIRCRAFT ATTITUDE

	Degrees	Radians
Bank u =	60°	1.04720
Pitch v =	-15°	-0.26180
Hdg w =	330°	5.75959

	Hdg	Pitch	Bank
$\cos(\theta/2)$:	-0.96593	0.99144	0.86603
$\sin(\theta/2)$:	0.25882	-0.13053	0.50000

Quaternion	
r_0	-0.84625
r_1	0.15919
r_2	0.23749
r_3	-0.44957

CAMERA SWIVEL

	Degrees	Radians
Roll u =	0°	0.00000
Tilt v =	20°	0.34907
Yaw w =	-35°	-0.61087

	Yaw	Tilt	Roll
$\cos(\theta/2)$:	0.95372	0.98481	1.00000
$\sin(\theta/2)$:	-0.30071	0.17365	0.00000

Quaternion	
s_0	0.93923
s_1	-0.29614
s_2	0.16561
s_3	0.05222

QUATERNION MULTIPLICATION

$t = r * s$

t_0	-0.76354
t_1	0.31326
t_2	-0.04192
t_3	-0.56313

EULER (TAIT-BRYAN) ANGLES FOR GOOGLE EARTH

			GE Tilt	90°
	Radians	Degrees	Degrees	Eqn.
ROLL	1.16090	**66.51°**		**(4)**
TILT	-0.42995	-24.63°	**65.37°**	**(3), (6)**
HDG	-0.49422	**-28.32°**		**(5)**

Euler (Tait-Bryan) Rotation Example
Compound Rotation

AIRCRAFT ATTITUDE

	Degrees	Radians
Bank u =	60°	1.04720
Pitch v =	-15°	-0.26180
Hdg w =	330°	5.75959

$\boldsymbol{R}_x(u) = $

1	0	0
0	0.50000	-0.86603
0	0.86603	0.50000

$\boldsymbol{R}_y(v) = $

0.96593	0	-0.25882
0	1	0
0.25882	0	0.96593

$\boldsymbol{R}_z(w) = $

0.86603	0.50000	0
-0.50000	0.86603	0
0	0	1

$\boldsymbol{R}_{FS} = \boldsymbol{R}_z$(Hdg) * $\boldsymbol{R}_y$(Pitch) * $\boldsymbol{R}_x$(Bank)

0.83652	0.05589	-0.54508
-0.48296	0.54508	-0.68530
0.25882	0.83652	0.48296

CAMERA SWIVEL

	Degrees	Radians
Roll u =	0°	0.00000
Tilt v =	20°	0.34907
Yaw w =	-35°	-0.61087

$\boldsymbol{R}_x(u) = $

1	0	0
0	1.00000	0.00000
0	0.00000	1.00000

$\boldsymbol{R}_y(v) = $

0.93969	0	0.34202
0	1	0
-0.34202	0	0.93969

$\boldsymbol{R}_z(w) = $

0.81915	0.57358	0
-0.57358	0.81915	0
0	0	1

$\boldsymbol{R}_{CAM} = \boldsymbol{R}_z$(Yaw) * $\boldsymbol{R}_y$(Tilt) * $\boldsymbol{R}_x$(Roll)

0.76975	0.57358	0.28017
-0.53899	0.81915	-0.19617
-0.34202	0.00000	0.93969

ROTATION MATRIX MULTIPLICATION

$\boldsymbol{R}_{FS\times CAM} = \boldsymbol{R}_{FS} * \boldsymbol{R}_{CAM}$

0.80022	0.52558	-0.28881
-0.43117	0.16949	-0.88621
-0.41683	0.83369	0.36225

EULER (TAIT-BRYAN) ANGLES FOR GOOGLE EARTH

	Radians	Degrees	GE Tilt Degrees	Eqn.
			GE Tilt	90°
ROLL	1.16090	**66.51°**		**(12)**
TILT	-0.42995	-24.63°	**65.37°**	**(11), (14)**
HDG	-0.49422	**-28.32°**		**(13)**

Excel Formulas: QUATERNION

	A	B	C	D	E	F	G	H
1		AIRCRAFT ATTITUDE				CAMERA SWIVEL		
2								
3			Degrees	Radians			Degrees	Radians
4		Bank u =	60°	=RADIANS(C4)		Roll u =	0°	=RADIANS(G4)
5		Pitch v =	-15°	=RADIANS(C5)		Tilt v =	20°	=RADIANS(G5)
6		Hdg w =	330°	=RADIANS(C6)		Yaw w =	-35°	=RADIANS(G6)
7								
8		Hdg	Pitch	Bank		Yaw	Tilt	Roll
9	cos(τ/2):	=COS(D6/2)	=COS(D5/2)	=COS(D4/2)		=COS(H6/2)	=COS(H5/2)	=COS(H4/2)
10	sin(τ/2):	=SIN(D6/2)	=SIN(D5/2)	=SIN(D4/2)		=SIN(H6/2)	=SIN(H5/2)	=SIN(H4/2)
11								
12		Quaternion				Quaternion		
13		r_0	=B9*C9*D9+B10*C10*D10			s_0	=F9*G9*H9+F10*G10*H10	
14		r_1	=B10*C9*D9-B9*C10*D10			s_1	=F10*G9*H9-F9*G10*H10	
15		r_2	=B9*C10*D9+B10*C9*D10			s_2	=F9*G10*H9+F10*G9*H10	
16		r_3	=B9*C9*D10-B10*C10*D9			s_3	=F9*G9*H10-F10*G10*H9	
17								
18				QUATERNION MULTIPLICATION				
19					$t = r * s$			
20				t_0	=C13*G13-C14*G14-C15*G15-C16*G16			
21				t_1	=C13*G14+C14*G13-C15*G16+C16*G15			
22				t_2	=C13*G15+C14*G16+C15*G13-C16*G14			
23				t_3	=C13*G16-C14*G15+C15*G14+C16*G13			
24								
25			EULER (TAIT-BRYAN) ANGLES FOR GOOGLE EARTH					
26								
27						GE Tilt	90°	
28				Radians	Degrees	Degrees	Eqn.	
29			ROLL	=ATAN2(E20^2+E21^2-E22^2-E23^2,2*(E20*E23+E21*E22))				
30			TILT	=-(ASIN(2*(E20*E22-E21*E23)))				
31			HDG	=(ATAN2(E20^2-E21^2-E22^2+E23^2, 2*(E20*E21+E22*E23)))				

Excel Formulas: EULER

	A	B	C	D	E	F	G	H
1		AIRCRAFT ATTITUDE				CAMERA SWIVEL		
2								
3			Degrees	Radians			Degrees	Radians
4		Bank u =	60°	=RADIANS(C4)		Roll u =	0°	=RADIANS(G4)
5		Pitch v =	-15°	=RADIANS(C5)		Tilt v =	20°	=RADIANS(G5)
6		Hdg w =	330°	=RADIANS(C6)		Yaw w =	-35°	=RADIANS(G6)
7								
8		1	0	0		1	0	0
9	$R_x(u)$ =	0	=COS(D4)	=-SIN(D4)	$R_x(u)$ =	0	=COS(H4)	=-SIN(H4)
10		0	=SIN(D4)	=COS(D4)		0	=SIN(H4)	=COS(H4)
11								
12		=COS(D5)	0	=SIN(D5)		=COS(H5)	0	=SIN(H5)
13	$R_y(v)$ =	0	1	0	$R_y(v)$ =	0	1	0
14		=-SIN(D5)	0	=COS(D5)		=-SIN(H5)	0	=COS(H5)
15								
16		=COS(D6)	=-SIN(D6)	0		=COS(H6)	=-SIN(H6)	0
17	$R_z(w)$ =	=SIN(D6)	=COS(D6)	0	$R_z(w)$ =	=SIN(H6)	=COS(H6)	0
18		0	0	1		0	0	1
19								
20		$R_{FS} = R_z$(Hdg) * R_y(Pitch) * R_x(Bank)				$R_{CAM} = R_z$(Yaw) * R_y(Tilt) * R_x(Roll)		
21								
22			=MMULT(B16:D18,MMULT(B12:D14,B8:D10))				=MMULT(F16:H18,MMULT(F12:H14,F8:H10))	
23								
24								
25				ROTATION MATRIX MULTIPLICATION				
26					$R_{FS \times CAM} = R_{FS} * R_{CAM}$			
27								
28			=MMULT(B16:D18,MMULT(B12:D14,MMULT(B8:D10,MMULT(F16:H18,MMULT(F12:H14,F8:H10)))))					
29								
30								
31			EULER (TAIT-BRYAN) ANGLES FOR GOOGLE EARTH					
32								
33						GE Tilt	90°	
34				Radians	Degrees	Degrees	Eqn.	
35			ROLL	=ATAN2(F29/COS(D36),E29/COS(D36))				
36			TILT	=ASIN(D29)				
37			HDG	=ATAN2(D27/COS(D36),D28/COS(D36))				

References:

There are scores of articles and forum threads related to 3D rotation on the web, but for the Flight Simulator – Google Earth application, the most succinct and relevant that I found are listed below.

Rose, David
http://danceswithcode.net/engineeringnotes/index.html

Slabaugh, Gregory
http://staff.city.ac.uk/~sbbh653/publications/euler.pdf

CH Robotics, Inc.
http://www.chrobotics.com/library

Baker, Martin
http://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/index.htm

McLellan, James
http://www.mclellansys.com/article_2.htm

and several Wikipedia entries such as
https://en.wikipedia.org/wiki/Euler_angles