SENTINEL-1 IPF: ATTITUDE QUATERNIONS USAGE

1. INTRODUCTION

The purpose of this note is to propose a sequence of EO CFI function calls to be implemented in the Sentinel-1 IPF in order to retrieve the quaternion attitude data from the POD Attitude Restituted files and SAR Source Packets (SSP).

1.1 Change History

Issue	Change Description
1.0	First Issue
1.1	Minor corrections
1.2	-Quaternion multiplication corrected (section 3.7.4) -Minor clarifications (sections 3.5 and 4.2) -Added Annex with definition of azimuth and elevation convention and EO CFI Sentinel-1 Nominal Attitude Frame (section 5)
2.0	-SAR Quaternions defined wrt Geocentric Mean of 2000 frame. -The Earth-Fixed/Geocentric Mean of 2000 (TBD) has been removed. -Added the steps to calculate the attitude rotation angles (sections 3.11 and 4.7) -EOCFI relevant conventions added to Annex B (section 6)
2.1	-Update proposed implementation for matrix calculation in Step 11.1 (section 3) and Step 6.1 (section 4). See Note for details. -Added numerical example in Steps 5.1 to 5.4 (Test Example sub-section) to illustrate the conversion from SSP quaternion to EOCFI quaternion with an actual test data sample -Remove comment about transposing SAR quaternion in note after Step 5.1 (section 3)

1.2 Distribution List

Project/Unit	Name	Project/Unit	Name	Project/Unit	Name
Sentinel-1	N. Miranda	Sentinel-1	P. Deghaye	System Support	B. Duesmann
System Support	M. Piñol Solé				

1.3 Reference Documents

[RD 01] EO CFI Software - EO Data Handling – Ref. EO-MA-DMS-GS-0007 – Issue 4.7 – 28/03/2014
[RD 02] EO CFI Software - EO Lib – Ref. EO-MA-DMS-GS-0003 – Issue 4.7 – 28/03/2014
[RD 03] EO CFI Software - EO Orbit – Ref. EO-MA-DMS-GS-0004 – Issue 4.7 – 28/03/2014
[RD 04] EO CFI Software - EO Pointing User Manual– Ref. EO-MA-DMS-GS-0005 – Issue 4.7 – 28/03/2014
[RD 05] GMES Sentinel-1 System Requirements Document. Ref. S1-RS-ESA-SY-0001 – Issue 4.0 – 11/11/2011
[RD 06] Sentinel-1 Control System Design Report Ref. S1-RP-TASI-SC-0033. Issue 7 – 26/10/10
[RD 07] Sentinel-1 SAR Instrument: Definition of Roll Steering Law. Ref. S1-TN-ASD-PL-0018. Issue 2 – 07/13/14
[RD 08] PVT and SCA CSAR Ancillary data acquired during the test performed on the Avionics test Bench. E-mail from L. Galvagni, sent on 30th September – Subject: [<Thales Group Crypt and Share>] File available: CSAR Ancillary ECI Quaternion ATB Test Data

2. ASSUMPTIONS

The following assumptions have been considered:

- Both the POD Restituted Orbit files and the internal orbit files generated from the SAR Source Packets are assumed to be compliant with the EO CFI restituted orbit file format (see [RD 01])
- Both the POD Restituted Attitude files and the internal attitude files generated from the SAR Source Packets are assumed to be compliant with the EO CFI attitude file format (see [RD 01])
- The attitude quaternions in the attitude files represent the rotation from Geocentric Mean of 2000 (if internal attitude files) or Earth-Fixed (if POD Restituted Attitude files) to Satellite Attitude Frame.
- The POD Restituted attitude files follow the quaternion and axes convention used by EO CFI (see Annex B).

3. DATA FROM SAR SOURCE PACKETS: OUTLINE OF THE CALLING SEQUENCE

3.1 Step 1

Read orbit state vectors (PVT) and attitude quaternions (QT) from the SAR Source Packets. The PVT and the QT may have different time stamps.

- satellite position and velocity vectors in Earth-Fixed Coordinate System at a given time t_i : r_{EF}, v_{EF}
- quaternion from Geocentric Mean of 2000 Frame to Satellite Attitude Frame at a time t_j :

  $Q_{GM\,2000 \rightarrow SATATT}$

In order to cope with the different time stamps, the PVT data has to be read and written to an internal EO CFI orbit file and the QT data has to be read and written (after transformation) to internal EO CFI attitude file.

3.2 Step 2

Obtain time correlation information (for example, from IERS Bulletin A or B). This time correlation information (UTC-TAI, UTC-UT1) will be used to write the internal EO CFI orbit file, together with the position and velocity vectors in Earth-Fixed CS.

3.3 Step 3

Write the time t_i and satellite position and velocity r_{EF}, v_{EF} to internal EO CFI orbit file.
EO CFI hint: Call function xd_write_orbit_file ([RD 01])

3.4 Step 4

Initialize the model_id. Initialize the time_id using internal EO CFI orbit file, IERS Bulletin A or B and the orbit_id using the internal EO CFI orbit file.
EO CFI hint: Call functions xl_model_init (mode default), xl_time_ref_init_file (mode AUTO) and xo_orbit_init_file (mode AUTO). See [RD 02] and [RD 03].

3.5 Step 5

Convert the quaternion from Geocentric Mean of 2000 Frame to Satellite Attitude Frame to a quaternion from Geocentric Mean of 2000 Frame to EO CFI Satellite Attitude Frame.
Details provided in steps 5.1 to 5.4 below.

3.5.1 Step 5.1

Re-arrange the elements of the input quaternion such that q_3 is the real part and (q_0, q_1, q_2) the vector part, see section 6.6 b).

$$q_0^{EOCFI} = q_1$$
$$q_{11}^{EOCFI} = q_2$$
$$q_2^{EOCFI} = q_3$$
$$q_3^{EOCFI} = q_0$$

Note: It is assumed that the quaternion in the SAR Source Packet represents the transformation from Geocentric Mean of 2000 to Satellite Attitude Frame, so the corresponding matrix would transform a vector in Geocentric Mean of 2000 into a vector in Satellite Attitude Frame (see Section 6).

3.5.1.1 *Proposed Implementation*

```
/* STEP 5.1: re-assign, q0 real --> q[3] in EO CFI convention*/
q[0] = q1;
q[1] = q2;
q[2] = q3;
q[3] = q0;
```

3.5.1.2 *Test Example*

A quaternion sample has been extracted from SCA CSAR Ancillary data acquired during the test performed on the Avionics test Bench (see [RD 08]):

```
/* Quaternion --> q0 = scalar part */
q0 = -0.3229468762874603272;
q1 = -0.9336623549461364746;
q2 = 0.02849436365067958832;
q3 = -0.1522108763456344604;
```

Output of Step 5.1 (re-assign quaternion components):

```
q0 = -0.9336623549461364746;
q1 = 0.02849436365067958832;
q2 = -0.1522108763456344604;
q3 = -0.3229468762874603272;
```

3.5.2 Step 5.2

Transform the quaternion obtained after Step 5.1 to a matrix.
Since the SAR Source Packet quaternion represents the transformation from Geocentric Mean of 2000 to Satellite Attitude Frame, then the matrix obtained would represent the rotation $M_{GM\,2000 \rightarrow SATATT}$, which transforms a vector in Geocentric Mean of 2000 into a vector in Satellite Attitude Frame

EO CFI hint: Call function xl_quaternions_to_vectors (see [RD 02]) and assign the output vectors of the function to the columns of a matrix:

$$Q = [\ q_0 \quad q_1 \quad q_2 \quad q_3\ ] \Leftrightarrow M = \begin{bmatrix} u_x^1 & u_y^1 & u_z^1 \\ u_x^2 & u_y^2 & u_z^2 \\ u_x^3 & u_y^3 & u_z^3 \end{bmatrix}$$

3.5.2.1 Proposed Implementation

```
/* STEP 5.2: Transform to matrix */
  status = xl_quaternions_to_vectors(q, ux_vec, uy_vec, uz_vec, xl_ierr);
  if (status != XL_OK)
    {
     func_id = XL_QUATERNIONS_TO_VEC_ID;
     xl_get_msg(&func_id, xl_ierr, &n, msg);
     xl_print_msg(&n, msg);
     if (status <= XL_ERR) return(XL_ERR);
  }

 /* Assign to matrix */
   for(i=0;i<3;i++){
       matrix_aux[i][0] = ux_vec[i];
       matrix_aux[i][1] = uy_vec[i];
       matrix_aux[i][2] = uz_vec[i];
  }
```

3.5.2.2 Test Example

Output of Step 5.2 (convert quaternion to matrix):

```
 0.952039848  0.045103818  0.302631414
-0.151520260 -0.789786806  0.594372284
 0.265822757 -0.611720890 -0.745074369
```

3.5.3 Step 5.3

Re-assign the rows of the resulting attitude matrix, to map the axes convention from Satellite Attitude Frame to EO CFI Satellite Attitude Frame (see Section 6.6. a))

According to Section 6.2:

$$P_{SATATT \rightarrow GM\,2000} = [\ \vec{X}^{GM\,2000}_{SATATT} \quad \vec{Y}^{GM\,2000}_{SATATT} \quad \vec{Z}^{GM\,2000}_{SATATT}\ ]$$

where

$+\vec{X}^{GM\,2000}_{SATATT}$ is the unitary direction vector along X-axis of Frame Satellite Attitude Frame (expressed in Geocentric Mean of 2000)

$+\vec{Y}^{GM\,2000}_{SATATT}$ is the unitary direction vector along Y-axis of Frame Satellite Attitude Frame (expressed in Geocentric Mean of 2000)

$+\vec{Z}^{GM\,2000}_{SATATT}$ is the unitary direction vector along Z-axis of Frame Satellite Attitude Frame (expressed in Geocentric Mean of 2000)

Then the rotation matrix from Step 7.2 would correspond to the transposed matrix:

$$M_{GM\,2000 \rightarrow SATATT} = \begin{bmatrix} \vec{X}_{SATATT} \\ \vec{Y}_{SATATT} \\ \vec{Z}_{SATATT} \end{bmatrix}$$

Using the relation between the axes definition of the Satellite Attitude Frame in [RD 06] and the EO CFI Satellite Attitude Frame (established in Section 6.6 a)):

$$+\vec{X}_{SATATT}^{EOCFI} = -\vec{Y}_{SATATT}$$
$$+\vec{Y}_{SATATT}^{EOCFI} = -\vec{X}_{SATATT}$$
$$+\vec{Z}_{SATATT}^{EOCFI} = -\vec{Z}_{SATATT}$$

Then to be compliant with the EO CFI Satellite Attitude Frame definition we just need to re-assign the rows of the resulting attitude matrix from Step 5.2:

$$M_{GM\,2000\rightarrow SATATT} = \begin{bmatrix} \vec{X}_{SATATT} \\ \vec{Y}_{SATATT} \\ \vec{Z}_{SATATT} \end{bmatrix} \Rightarrow N_{GM\,2000\rightarrow EOCFI\_SATATT} = \begin{bmatrix} -\vec{Y}_{SATATT}^{EOCFI} \\ -\vec{X}_{SATATT}^{EOCFI} \\ -\vec{Z}_{SATATT}^{EOCFI} \end{bmatrix}$$

3.5.3.1 Proposed Implementation

```
/* STEP 5.3: Re-assign rows to go from Sentinel-1 Attitude frame axes convention to EO
CFI Satellite Attitude Frame definition */
for (i=0;i<3;i++)
{
        matrix_aux2[0][i] = -matrix_aux[1][i];
        matrix_aux2[1][i] = -matrix_aux[0][i];
        matrix_aux2[2][i] = -matrix_aux[2][i];
}
```

3.5.3.2 Test Example

Output of Step 5.3 (re-assign rows to map S1 to EO CFI satellite attitude frame axes):

```
 0.151520260  0.789786806 -0.594372284
-0.952039848 -0.045103818 -0.302631414
-0.265822757  0.611720890  0.745074369
```

3.5.4 Step 5.4

Calculate the quaternion corresponding to the transformation matrix from Geocentric Mean of 2000 to EO CFI Satellite Attitude Frame $N_{GM\,2000\rightarrow EOCFI\_SATATT}$ (from Step 5.3)

EO CFI hint: Call function xl_vectors_to_quaternions (see [RD 02]) assigning the columns of the matrix to the input vectors u_x, u_y, u_z:

$$N_{GM\,2000\rightarrow EOCFI\_SATATT} = \begin{bmatrix} u_x^1 & u_y^1 & u_z^1 \\ u_x^2 & u_y^2 & u_z^2 \\ u_x^3 & u_y^3 & u_z^3 \end{bmatrix}$$

The quaternion will be: $S = \begin{bmatrix} s_0^{EOCFI} & s_1^{EOCFI} & s_2^{EOCFI} & s_3^{EOCFI} \end{bmatrix}$ with s_3^{EOCFI} the real part.

3.5.4.1 Proposed Implementation

```
/* STEP 5.4: Assign matrix to vectors and calculate the EO CFI quaternion */

for(i=0;i<3;i++){
  ux_vec[i] = matrix_aux2[i][0];
```

```
 uy_vec[i] = matrix_aux2[i][1];
 uz_vec[i] = matrix_aux2[i][2];
 }

status = xl_vectors_to_quaternions(ux_vec, uy_vec, uz_vec, q_cfi, xl_ierr);
 if (status != XL_OK)
   {
    func_id = XL_VEC_TO_QUATERNIONS_ID;
    xl_get_msg(&func_id, xl_ierr, &n, msg);
    xl_print_msg(&n, msg);
    if (status <= XL_ERR) return(XL_ERR);
    }
```

3.5.4.2 *Test Example*

Output of Step 5.4 (convert matrix to quaternion):

```
q0 = -0.335987242547;
q1 = 0.120728573839;
q2 = 0.640050374327;
q3 = 0.680347486678;
```

3.6 Step 6

Write time t_j and quaternion data $S = \left[\begin{array}{cccc} s_0^{EOCFI} & s_1^{EOCFI} & s_2^{EOCFI} & s_3^{EOCFI} \end{array} \right]$ to internal EO CFI attitude quaternion file.
EO CFI hint: Create fixed header structure. Call function xd_write_att (see [RD 01]).

3.7 Step 7

Initialize the Satellite Nominal Attitude with the EO CFI Zero-Doppler Attitude Frame.
EO CFI hint: Call xp_sat_nominal_att_init_model with mode SENTINEL-1 and setting the roll steering parameters to zero (so only the Zero-Doppler attitude is applied as nominal law, see Section 5.1.1):

```
model_param[0] = 0.0060611;
model_param[1] =  -0.729211585E-4;
model_param[2] = 0.0;
model_param[3] = 0.0;
model_param[4] = 0.0;
model_param[5] = 0.0;
model_param[6] = 0.0;
model_param[7] = 0.0;
model_param[8] = 0.0;
model_param[9] = 0.0;
model_param[10] = 0.0;
model_param[11] = 0.0;
model_param[12] = 0.0;
model_param[13] = 0.0;
```

See [RD 04].

3.8 Step 8

Initialize the Satellite Attitude using the EO CFI attitude quaternion file.
EO CFI hint: Call function xp_sat_att_init_file (see [RD 04]).

3.9 Step 9 (optional)

If the instrument is misaligned wrt Satellite Attitude Frame, initialize the Instrument Attitude using the EO CFI instrument frame initialization function.
EO CFI hint: For example, call function xp_instr_att_angle_init or xp_instr_att_matrix_init (see [RD 04]).

3.10 Step 10

Compute the satellite position and velocity vectors in Earth-Fixed at a given time t . The output state vectors will be used in Step 11.
EO CFI hint: Call function xo_osv_compute (see [RD 03])

3.11 Step 11

Calculate the attitude rotation angles between EOCFI Satellite Nominal Frame (set to EO CFI Zero-Doppler Attitude Frame, see Step 7) and EOCFI Satellite Attitude Frame (i.e. Roll Steering Law angles).
Details provided in steps 11.1 to 11.2 below.

3.11.1 Step 11.1

Compute the transformation matrix $P_{ZDOP \rightarrow EOCFI\_SATATT}$ that transforms a vector in EO CFI Zero-Doppler Attitude Frame (= EO CFI Satellite Nominal Attitude Frame) into a vector in Satellite Attitude Frame for time t : $u_{EOCFI\_SATATT} = P_{ZDOP \rightarrow EOCFI\_SATATT} \cdot u_{ZDOP}$

Then, according to Section 6.2, $P_{ZDOP \rightarrow EOCFI\_SATATT} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}$

with
$X = (x_1, x_2, x_3)$ the unitary direction vector along the X-axis of the EO CFI Zero-Doppler Attitude Frame (expressed in EO CFI Satellite Attitude Frame)
$Y = (y_1, y_2, y_3)$ the unitary direction vector along the Y-axis of the EO CFI Zero-Doppler Attitude Frame expressed in EO CFI Satellite Attitude Frame)
$Z = (z_1, z_2, z_3)$ the unitary direction vector along the Z-axis of the EO CFI Zero-Doppler Attitude Frame (expressed in EO CFI Satellite Nominal Frame)

EO CFI hint: Call function xp_change_frame three times (in mode direction, input frame set to Satellite Nominal Attitude, output frame set to Satellite Attitude Frame) with input unitary direction vectors $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$. The output vectors correspond to the columns of the transformation matrix from Satellite Nominal Attitude Frame to Satellite Attitude Frame. The satellite position and velocity vectors at time t will be used as input. See [RD 04].
Note: In order to circumvent an anomaly in EO CFI v4.7, the implementation proposed below makes use of an intermediate frame (e.g. Geocentric Mean of 2000) to obtain two matrices and build the transformation matrix from Satellite Nominal Attitude Frame to Satellite Attitude Frame with the product matrix as follows: $C_{ZDOP \rightarrow EOCFI\_SATATT} = A_{GM\,200 \rightarrow EOCFI\_SATATT} \cdot B_{ZDOP \rightarrow GM\,2000}$

3.11.1.1 Proposed Implementation

```
/* STEP 11.1a: Calculate rotation matrix between Geocentric Mean of 2000 Frame and
Satellite Attitude Frame (initialized from internal attitude file) */
mode                = XP_MODE_FLAG_DIRECTION;
deriv               = XP_NO_DER;
```

```
  frame_flag_input    = XP_FRAME_FLAG_EXT;
  frame_id_input      = XP_GM2000;
   frame_flag_output  = XP_FRAME_FLAG_SAT;
   frame_id_output    = XP_SAT_ATT;

   vec_input[0]             =   0.0;
  vec_input[1]              =   0.0;
  vec_input[2]              =   0.0;
  vec_rate_input[0]         =   0.0;
  vec_rate_input[1]         =   0.0;
  vec_rate_input[2]         =   0.0;
  vec_rate_rate_input[0]    =   0.0;
  vec_rate_rate_input[1]    =   0.0;
  vec_rate_rate_input[2]    =   0.0;

  for (j=0;j<3;j++)
  {
        vec_input[j] = 1.0;

        status = xp_change_frame(&sat_id,&model_id,&time_id,
                                 &sat_nom_trans_id,&sat_trans_id,&instr_trans_id,
                                 &mode,
                                 &frame_flag_input,&frame_id_input,
                                 &frame_flag_output,&frame_id_output,
                                 &time_ref,&time,
                                 pos,vel,acc,&deriv,
                                 vec_input,vec_rate_input,vec_rate_rate_input,
                                 /* output */
                                 vec_output,vec_rate_output,vec_rate_rate_output,
                                 xp_ierr);

        if (status != XP_OK)
        {
              func_id = XP_CHANGE_FRAME_ID;
              xp_get_msg(&func_id, xp_ierr, &n, msg);
              xp_print_msg(&n, msg);
              if (status <= XP_ERR) return(XP_ERR);
        }

        matrix_j2000_att_frame[0][j] = vec_output[0];
        matrix_j2000_att_frame[1][j] = vec_output[1];
        matrix_j2000_att_frame[2][j] = vec_output[2];

        vec_input[j] = 0.0;
}
  /* STEP 11.1b: Calculate rotation matrix between Satellite Nominal Frame (initialized
  as Zero-Doppler frame) and Geocentric Mean of 2000 */
   mode               = XP_MODE_FLAG_DIRECTION;
  deriv               = XP_NO_DER;
  frame_flag_input    = XP_FRAME_FLAG_SAT;
  frame_id_input      = XP_SAT_NOMINAL_ATT;
   frame_flag_output  = XP_FRAME_FLAG_EXT;
   frame_id_output    = XL_GM2000;
```

```
vec_input[0]              =   0.0;
vec_input[1]              =   0.0;
vec_input[2]              =   0.0;
vec_rate_input[0]         =   0.0;
vec_rate_input[1]         =   0.0;
vec_rate_input[2]         =   0.0;
vec_rate_rate_input[0]    =   0.0;
vec_rate_rate_input[1]    =   0.0;
vec_rate_rate_input[2]    =   0.0;


for (j=0;j<3;j++)
{
       vec_input[j] = 1.0;

       status = xp_change_frame(sat_id,model_id,time_id,
                                sat_nom_trans_id,sat_trans_id,instr_trans_id,
                                &mode,
                                &frame_flag_input,&frame_id_input,
                                &frame_flag_output,&frame_id_output,
                                &time_ref,time,
                                pos,vel,acc,&deriv,
                                vec_input,vec_rate_input,vec_rate_rate_input,
                                /* output */
                                vec_output,vec_rate_output,vec_rate_rate_output,
                                xp_ierr);


       if (status != XP_OK)
       {
              func_id = XP_CHANGE_FRAME_ID;
              xp_get_msg(&func_id, xp_ierr, &n, msg);
              xp_print_msg(&n, msg);
              if (status <= XP_ERR) return(XP_ERR);
       }

       matrix_zdoppler_frame_j2000[0][j] = vec_output[0];
       matrix_zdoppler_frame_j2000[1][j] = vec_output[1];
       matrix_zdoppler_frame_j2000[2][j] = vec_output[2];

       vec_input[j] = 0.0;
}

/* STEP 11.1c: Calculate product matrix to obtain transformation between Satellite
Nominal Frame (initialized as Zero-Doppler frame) and Satellite Attitude Frame
(initialized from internal attitude file) */

for (i=0;i<3;i++){
        for (j=0;j<3;j++){
            sum = 0.0;
            for (k=0;k<3;k++){
                sum += matrix_j2000_att_frame[i][k]*matrix_zdoppler_frame_j2000[k][j];
                matrix_zdoppler_att_frame [i][j] = sum;
            }
```

```
        }
    }
```

3.11.2 Step 11.2

Calculate the attitude rotation angles from the matrix obtained in Step 11.1.

EO CFI hint: Call function xl_matrix_to_euler (see [RD 02]).

3.11.2.1 Proposed Implementation

```
/* STEP 11.2: Extract attitude rotation angles from rotation matrix between Satellite
Nominal Frame (initialized as Zero-Doppler frame) and Satellite Attitude Frame
(initialized from internal attitude file). Output angles in degrees. */

status = xl_matrix_to_euler(matrix_zdoppler_att_frame, angles_att, xl_ierr);
if (status != XL_OK)
{
  func_id = XL_MATRIX_TO_EULER_ID;
  xl_get_msg(&func_id, xl_ierr, &n, msg);
  xl_print_msg(&n, msg);
  if (status <= XL_ERR) return(XL_ERR);
}
```

3.12 Step 12

After the initialization steps, it is possible to compute satellite, attitude and target data for a given time.
EO CFI hint: E.g. call functions xo_osv_compute + xo_osv_compute_extra (see [RD 03]), xp_attitude_compute, xp_target_xxx, xp_target_extra_xxx (see [RD 04]).

4. DATA FROM RESTITUTED ATTITUDE FILE: OUTLINE OF THE CALLING SEQUENCE

4.1 Step 1

Initialize the model_id. Initalize the time_id and orbit_id using the Restituted orbit file.
EO CFI hint: Call functions xl_model_init (mode default), xl_time_ref_init_file (mode AUTO) and xo_orbit_init_file (mode AUTO). See [RD 02] and [RD 03].

4.2 Step 2

Initialize the Satellite Nominal Attitude with the EO CFI Zero-Doppler Attitude Frame.
EO CFI hint: Call xp_sat_nominal_att_init_model with mode SENTINEL-1 and setting the roll steering parameters to zero (so only the Zero-Doppler attitude is applied as nominal law, see Section 5.1.1):

```
model_param[0] = 0.0060611;
model_param[1] =  -0.729211585E-4;
model_param[2] = 0.0;
model_param[3] = 0.0;
model_param[4] = 0.0;
model_param[5] = 0.0;
model_param[6] = 0.0;
model_param[7] = 0.0;
model_param[8] = 0.0;
model_param[9] = 0.0;
model_param[10] = 0.0;
model_param[11] = 0.0;
model_param[12] = 0.0;
model_param[13] = 0.0;
```

See [RD 04].

4.3 Step 3

Initialize the Satellite Attitude using the POD Restituted attitude quaternion file.
EO CFI hint: Call function xp_sat_att_init_file (see [RD 04]).

Note: It is assumed that the POD Restituted attitude file follows the quaternion and axes convention used by EO CFI, so it is not required any intermediate conversion.

4.4 Step 4 (optional)

If the instrument is misaligned wrt Satellite Attitude Frame, initialize the Instrument Attitude using the EO CFI instrument frame initialization function.
EO CFI hint: For example, call function xp_instr_att_angle_init or xp_instr_att_matrix_init (see [RD 04]).

4.5 Step 5

Compute the satellite position and velocity vectors in Earth-Fixed at a given time t. The output state vectors will be used in Step 6.

EO CFI hint: Call function xo_osv_compute (see [RD 03])

4.6 Step 6

Calculate the attitude rotation angles between EOCFI Satellite Nominal Frame (set to EO CFI Zero-Doppler Attitude Frame, see Step 5) and EOCFI Satellite Attitude Frame (i.e. Roll Steering Law angles).

Details provided in steps 6.1 to 6.2 below.

4.6.1 Step 6.1

Compute the transformation matrix $P_{ZDOP \to EOCFI\_SATATT}$ that transforms a vector in EO CFI Zero-Doppler Attitude Frame (= EO CFI Satellite Nominal Attitude Frame) to a vector in Satellite Attitude Frame for time t : $u_{EOCFI\_SATATT} = P_{ZDOP \to EOCFI\_SATATT} \cdot u_{ZDOP}$

Then, according to Section 6.2, $P_{ZDOP \to EOCFI\_SATATT} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}$

with

$X = (x_1, x_2, x_3)$ the unitary direction vector along the X-axis of the EO CFI Zero-Doppler Attitude Frame (expressed in EO CFI Satellite Attitude Frame)

$Y = (y_1, y_2, y_3)$ the unitary direction vector along the Y-axis of the EO CFI Zero-Doppler Attitude Frame expressed in EO CFI Satellite Attitude Frame)

$Z = (z_1, z_2, z_3)$ the unitary direction vector along the Z-axis of the EO CFI Zero-Doppler Attitude Frame (expressed in EO CFI Satellite Attitude Frame)

EO CFI hint: Call function xp_change_frame three times (in mode direction, input frame set to Satellite Nominal Attitude, output frame set to Satellite Attitude Frame) with input unitary direction vectors $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$. The output vectors correspond to the columns of the transformation matrix from Satellite Nominal Attitude Frame to Satellite Attitude Frame. The satellite position and velocity vectors at time t will be used as input. See [RD 04].

Note: In order to circumvent an anomaly in EO CFI v4.7, the implementation proposed below makes use of an intermediate frame (e.g. Geocentric Mean of 2000) to obtain two matrices and build the transformation matrix from Satellite Nominal Attitude Frame to Satellite Attitude Frame with the product matrix as follows: $C_{ZDOP \to EOCFI\_SATATT} = A_{GM\,200 \to EOCFI\_SATATT} \cdot B_{ZDOP \to GM\,2000}$

4.6.1.1 Proposed Implementation

```
/* STEP 6.1a: Calculate rotation matrix between Geocentric Mean of 2000 Frame and
Satellite Attitude Frame (initialized from internal attitude file) */
mode                = XP_MODE_FLAG_DIRECTION;
deriv               = XP_NO_DER;
frame_flag_input    = XP_FRAME_FLAG_EXT;
frame_id_input      = XP_GM2000;
frame_flag_output   = XP_FRAME_FLAG_SAT;
frame_id_output     = XP_SAT_ATT;

vec_input[0]            =   0.0;
vec_input[1]            =   0.0;
vec_input[2]            =   0.0;
vec_rate_input[0]       =   0.0;
vec_rate_input[1]       =   0.0;
vec_rate_input[2]       =   0.0;
vec_rate_rate_input[0]  =   0.0;
vec_rate_rate_input[1]  =   0.0;
vec_rate_rate_input[2]  =   0.0;
```

```
  for (j=0;j<3;j++)
  {
        vec_input[j] = 1.0;

        status = xp_change_frame(&sat_id,&model_id,&time_id,
                                 &sat_nom_trans_id,&sat_trans_id,&instr_trans_id,
                                 &mode,
                                 &frame_flag_input,&frame_id_input,
                                 &frame_flag_output,&frame_id_output,
                                 &time_ref,&time,
                                 pos,vel,acc,&deriv,
                                 vec_input,vec_rate_input,vec_rate_rate_input,
                                 /* output */
                                 vec_output,vec_rate_output,vec_rate_rate_output,
                                 xp_ierr);


        if (status != XP_OK)
        {
              func_id = XP_CHANGE_FRAME_ID;
              xp_get_msg(&func_id, xp_ierr, &n, msg);
              xp_print_msg(&n, msg);
              if (status <= XP_ERR) return(XP_ERR);
        }

        matrix_j2000_att_frame[0][j] = vec_output[0];
        matrix_j2000_att_frame[1][j] = vec_output[1];
        matrix_j2000_att_frame[2][j] = vec_output[2];

        vec_input[j] = 0.0;
}
  /* STEP 6.1b: Calculate rotation matrix between Satellite Nominal Frame (initialized as
  Zero-Doppler frame) and Geocentric Mean of 2000 */
   mode                = XP_MODE_FLAG_DIRECTION;
  deriv                = XP_NO_DER;
  frame_flag_input     = XP_FRAME_FLAG_SAT;
  frame_id_input       = XP_SAT_NOMINAL_ATT;
   frame_flag_output   = XP_FRAME_FLAG_EXT;
   frame_id_output     = XL_GM2000;

  vec_input[0]             =   0.0;
  vec_input[1]             =   0.0;
  vec_input[2]             =   0.0;
  vec_rate_input[0]        =   0.0;
  vec_rate_input[1]        =   0.0;
  vec_rate_input[2]        =   0.0;
  vec_rate_rate_input[0]   =   0.0;
  vec_rate_rate_input[1]   =   0.0;
  vec_rate_rate_input[2]   =   0.0;


  for (j=0;j<3;j++)
  {
```

```
        vec_input[j] = 1.0;

        status = xp_change_frame(sat_id,model_id,time_id,
                                 sat_nom_trans_id,sat_trans_id,instr_trans_id,
                                 &mode,
                                 &frame_flag_input,&frame_id_input,
                                 &frame_flag_output,&frame_id_output,
                                 &time_ref,time,
                                 pos,vel,acc,&deriv,
                                 vec_input,vec_rate_input,vec_rate_rate_input,
                                 /* output */
                                 vec_output,vec_rate_output,vec_rate_rate_output,
                                 xp_ierr);


        if (status != XP_OK)
        {
               func_id = XP_CHANGE_FRAME_ID;
               xp_get_msg(&func_id, xp_ierr, &n, msg);
               xp_print_msg(&n, msg);
               if (status <= XP_ERR) return(XP_ERR);
        }

        matrix_zdoppler_frame_j2000[0][j] = vec_output[0];
        matrix_zdoppler_frame_j2000[1][j] = vec_output[1];
        matrix_zdoppler_frame_j2000[2][j] = vec_output[2];

        vec_input[j] = 0.0;
}

/* STEP 6.1c: Calculate product matrix to obtain transformation between Satellite
Nominal Frame (initialized as Zero-Doppler frame) and Satellite Attitude Frame
(initialized from internal attitude file) */

for (i=0;i<3;i++){
        for (j=0;j<3;j++){
            sum = 0.0;
            for (k=0;k<3;k++){
                sum += matrix_j2000_att_frame[i][k]*matrix_zdoppler_frame_j2000[k][j];
                matrix_zdoppler_att_frame [i][j] = sum;
            }
        }
    }
```

4.6.2 Step 6.2

Calculate the attitude rotation angles from the matrix obtained in Step 6.1.

EO CFI hint: Call function xl_matrix_to_euler (see [RD 02])

4.6.2.1 Proposed Implementation

```
/* STEP 6.2: Extract attitude rotation angles from rotation matrix between Satellite
Nominal Frame (initialized as Zero-Doppler frame) and Satellite Attitude Frame
(initialized from internal attitude file). Output angles in degrees. */
```

```
status = xl_matrix_to_euler(matrix_zdoppler_att_frame, angles_att, xl_ierr);
if (status != XL_OK)
{
  func_id = XL_MATRIX_TO_EULER_ID;
  xl_get_msg(&func_id, xl_ierr, &n, msg);
  xl_print_msg(&n, msg);
  if (status <= XL_ERR) return(XL_ERR);
}
```

4.7 Step 4

After the initialization steps, it is possible to compute satellite, attitude and target data for a given time.
EO CFI hint: E.g. call functions xo_osv_compute + xo_osv_compute_extra (see [RD 03]), xp_attitude_compute, xp_target_xxx, xp_target_extra_xxx (see [RD 04]).

5. ANNEX A

5.1 EO CFI Sentinel-1 Satellite Nominal Attitude Frame

The EOCFI Sentinel-1 Satellite Nominal Attitude mode (initialized using xp_sat_nominal_att_init_model) is defined as follows:

$\vec{r}$ position vector in True of Date Reference Frame
$\vec{v}$ velocity vector in True of Date Reference Frame

$\vec{r}'$ position vector corrected for Earth's eccentricity:

$$\vec{r}' = \vec{r} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \beta \end{bmatrix} \cdot \vec{r}$$

where $\beta = 0.0060611$. This value is set with input parameter param_model[0]:

```
model_param[0] = 0.0060611;
```

$\vec{v}'$ velocity vector corrected for Earth's rotation:

$$\vec{v}' = \vec{v} + \begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix} \cdot \vec{r}$$

where $\omega = -0.729211585 \times 10^{-4}$ rad/s. This value is set with input parameter param_model[1]:

```
model_param[1] =  -0.729211585E-4;
```

Then

$$\vec{X} = \frac{-\vec{v}' \times \vec{r}'}{\left|-\vec{v}' \times \vec{r}'\right|}$$

$$\vec{Y} = \frac{-\vec{v}'}{\left|-\vec{v}'\right|}$$

$$\vec{Z} = \vec{X} \times \vec{Y}$$

The matrix $M = \begin{bmatrix} \vec{X} \\ \vec{Y} \\ \vec{Z} \end{bmatrix} = \begin{bmatrix} x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \\ z_0 & z_1 & z_2 \end{bmatrix}$ transforms a vector in True of Date to a vector in Zero-Doppler Attitude Frame (ZD), i.e.

$$x_{ZD} = M x_{ToD} .$$

The roll steering parameters are used to compute the roll steering rotation angle α. A rotation matrix is then applied to the EO CFI Zero-Doppler Attitude Frame to obtain the EO CFI Sentinel-1 Satellite Nominal Attitude Frame:

$$x_{SNAF} = R_Y(\alpha) \cdot x_{ZD}$$

The values defining the roll steering law are set through the input array param_model, array indexes from 2 to 13 (derived from values in [RD 07]:

```
model_param[2] = 29.450;
model_param[3] = 711.700;
model_param[4] = 0.05660;
model_param[5] = 707714.8;
model_param[6] = 8351.5;
model_param[7] = 8947.0;
model_param[8] = 23.32;
model_param[9] = 11.74;
model_param[10] = 3.1495;
model_param[11] = -1.5655;
model_param[12] = -3.1297;
model_param[13] = 4.7222;
```

5.1.1 Particular case: EO CFI Satellite Nominal Attitude set to Zero-Doppler Attitude

In the particular case in which the roll steering parameters are set to zero (as indicated in Section 3.7 and 4.2), the rotation matrix $R_Y(\alpha)$ becomes the identity matrix. This means that the resulting EO CFI Sentinel-1 Satellite Nominal Attitude Frame will correspond to the EO CFI Sentinel-1 Zero-Doppler Attitude Frame.

```
model_param[0] = 0.0060611;
model_param[1] =  -0.729211585E-4;
model_param[2] = 0.0;
model_param[3] = 0.0;
model_param[4] = 0.0;
model_param[5] = 0.0;
model_param[6] = 0.0;
model_param[7] = 0.0;
model_param[8] = 0.0;
model_param[9] = 0.0;
model_param[10] = 0.0;
model_param[11] = 0.0;
model_param[12] = 0.0;
model_param[13] = 0.0;
```

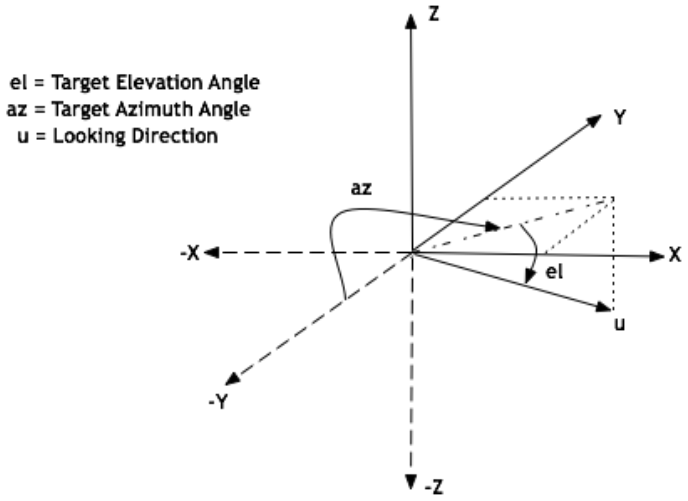
The initialization of the Zero-Doppler Attitude frame is needed in order to calculate the attitude rotation angles between the EO CFI Sentinel-1 Zero-Doppler frame (theoretical law) and the Satellite Attitude Frame (quaternions from the SAR Source Packets).

Note that this definition of the EO CFI Zero-Doppler Attitude Frame is in line with the Zero-Doppler Orbital Reference Frame definition in the SRD [RD 05], except for the axes naming convention.
If we look at the axes definition of the Zero-Doppler Orbital Reference Frame as per Sentinel-1 SRD [RD 05]) we observe that the relation between the Zero-Doppler Orbital Reference Frame (ZD_SRD) and the EO CFI Sentinel-1 Satellite Nominal Attitude Frame (SNAF) is:

$$+\vec{X}_{SNAF} = -\vec{T}_{ZD\_SRD}$$
$$+\vec{Y}_{SNAF} = -\vec{R}_{ZD\_SRD}$$
$$+\vec{Z}_{SNAF} = +\vec{L}_{ZD\_SRD}$$

5.2 EO CFI Azimuth and Elevation Angles Definition

The following convention for the target azimuth and elevation angles in Attitude Frame is assumed in the EO CFI:

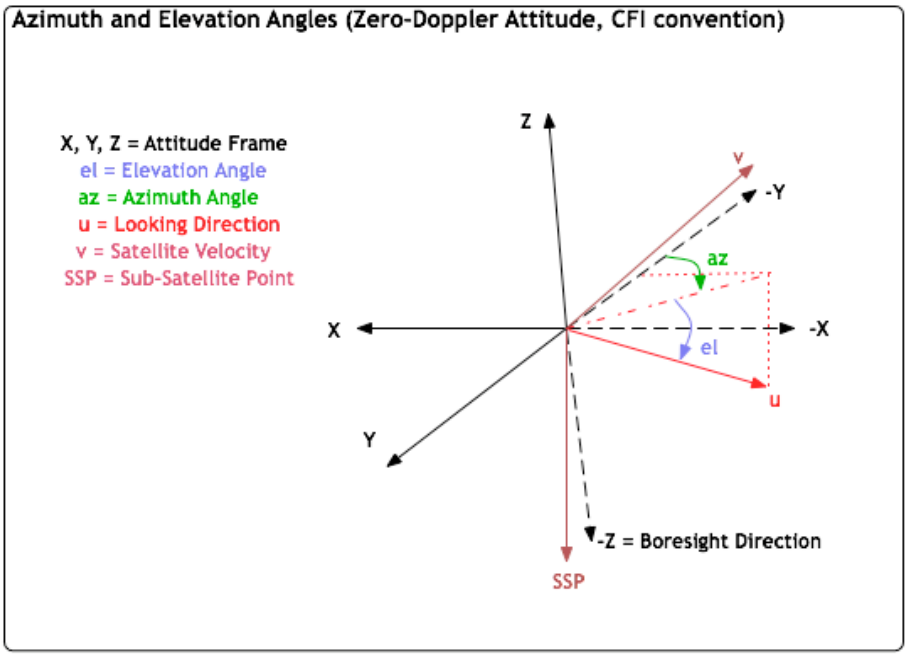


The Elevation Angle (allowed range between [-90, +90]) is positive when pointing downwards, e.g. $-\vec{Z} = +90°$ elevation, $+\vec{Z} = -90°$ elevation.

The Azimuth Angle (allowed range between [0, 360)) is positive from -Y towards -X, e.g. $+\vec{X} = +270°$ azimuth, $-\vec{Y} = 0°$ azimuth.

5.2.1 Example: Azimuth and Elevation Angles in EO CFI Sentinel-1 Zero-Doppler Attitude Frame

The figure below depicts the pointing azimuth and elevation angles on top of the Sentinel-1 Satellite Nominal Attitude Frame (assuming roll steering is not applied):



6. ANNEX B

6.1 EO CFI Quaternion Definition

The following quaternion definition is used:

$$Q = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}$$

with

$$q_0 = e_x \cdot \sin\left(\frac{\theta}{2}\right)$$

$$q_1 = e_y \cdot \sin\left(\frac{\theta}{2}\right)$$

$$q_2 = e_z \cdot \sin\left(\frac{\theta}{2}\right)$$

$$q_3 = \cos\left(\frac{\theta}{2}\right)$$

where (e_x, e_y, e_z) are the direction cosines of the rotation axis and θ is the rotation angle.

Then (q_0, q_1, q_2) corresponds to the vector part of the quaternion and q_3 is the scalar part of the quaternion.

6.2 EO CFI Rotation Matrix Definition

The rotation matrix $M_{A \to B}$ transforms a vector in Frame A into a vector in Frame B, i.e. $x_B = M_{A \to B} x_A$

where

$$M_{A \to B} = \begin{bmatrix} x_0 & y_0 & z_0 \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{bmatrix}$$

(x_0, x_1, x_2) is the unitary direction vector along X-axis of Frame A (expressed in Frame B)

(y_0, y_1, y_2) is the unitary direction vector along Y-axis of Frame A (expressed in Frame B)

(z_0, z_1, z_2) is the unitary direction vector along Z-axis of Frame A (expressed in Frame B)

6.3 Equivalence between Quaternion and Rotation Matrix in EO CFI

The quaternions in the attitude data files represent the transformation from a given reference frame (Earth-Fixed or Inertial) to Satellite Attitude Frame:

$$Q_{RF \to ATT} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}$$

The quaternion $Q_{RF \to ATT}$ is associated to the rotation matrix $M_{RF \to ATT}$ as follows:

$$Q_{RF\to ATT}=\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \Leftrightarrow M_{RF\to ATT}=\begin{bmatrix} q_0^2-q_1^2-q_2^2+q_3^2 & 2(q_0\cdot q_1+q_2\cdot q_3) & 2(q_0\cdot q_2-q_1\cdot q_3) \\ 2(q_0\cdot q_1-q_2\cdot q_3) & -q_0^2+q_1^2-q_2^2+q_3^2 & 2(q_1\cdot q_2+q_0\cdot q_3) \\ 2(q_0\cdot q_2+q_1\cdot q_3) & 2(q_1\cdot q_2-q_0\cdot q_3) & -q_0^2-q_1^2+q_2^2+q_3^2 \end{bmatrix}$$

where $M_{RF\to SATATT}$ transforms a vector in a given Reference Frame into a vector in Satellite Attitude Frame, i.e.

$x_{SATATT}=M_{RF\to SATATT}x_{RF}$

6.4 EO CFI Attitude Rotation Angles

Whenever a transformation is expressed as a sequence of rotations, the following expressions apply (the angle w is regarded positive):

$$R_X(w)=\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos w & \sin w \\ 0 & -\sin w & \cos w \end{bmatrix} R_Y(w)=\begin{bmatrix} \cos w & 0 & -\sin w \\ 0 & 1 & 0 \\ \sin w & 0 & \cos w \end{bmatrix} R_Z(w)=\begin{bmatrix} \cos w & \sin w & 0 \\ -\sin w & \cos w & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The following convention has been applied when using rotation angles roll-pitch-yaw to rotate one satellite related reference frame to another satellite related reference frame. In the particular case of the Satellite Nominal Attitude Frame and the Satellite (actual) Attitude Frame, we would have:

$$M_{SATNOM\to SATATT}=R_Z(yaw)\cdot R_X(-pitch)\cdot R_Y(-roll)=$$

$$=\begin{bmatrix} \cos(y)\cdot\cos(r)+\sin(r)\cdot\sin(y)\cdot\sin(p) & \sin(y)\cdot\cos(p) & \cos(y)\cdot\sin(r)-\cos(r)\cdot\sin(y)\cdot\sin(p) \\ -\sin(y)\cdot\cos(r)+\sin(r)\cdot\sin(p)\cdot\cos(y) & \cos(y)\cdot\cos(p) & -\sin(y)\cdot\sin(r)-\sin(p)\cdot\cos(y)\cdot\cos(r) \\ -\sin(r)\cdot\cos(p) & \sin(p) & \cos(r)\cdot\cos(p) \end{bmatrix}$$

where $M_{SATNOM\to SATATT}$ transforms a vector in Satellite Nominal Attitude Frame into a vector in Satellite Attitude Frame, i.e.

$x_{SATATT}=M_{SATNOM\to SATATT}x_{SATNOM}$

6.5 SAR Source Packet Quaternion Definition

The following quaternion definition is used:

$$Q=\begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}$$

with

$$q_0 = \cos\left(\frac{\theta}{2}\right)$$
$$q_1 = e_x \cdot \sin\left(\frac{\theta}{2}\right)$$
$$q_2 = e_y \cdot \sin\left(\frac{\theta}{2}\right)$$
$$q_3 = e_z \cdot \sin\left(\frac{\theta}{2}\right)$$

where (e_x, e_y, e_z) are the direction cosines of the rotation axis and θ is the rotation angle.

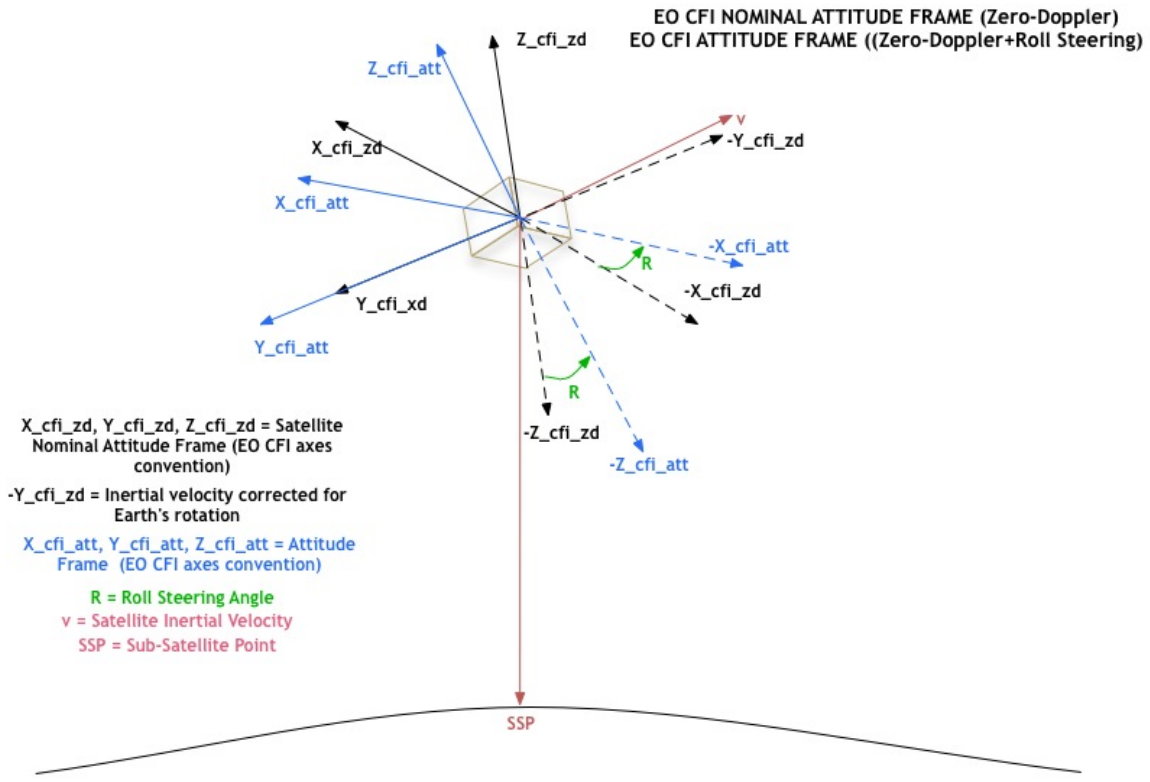
Then (q_1, q_2, q_3) corresponds to the vector part of the quaternion and q_0 is the scalar part of the quaternion.
The quaternion in the SAR Source Packet represents the transformation from Geocentric Mean of 2000 to Satellite Attitude Frame.

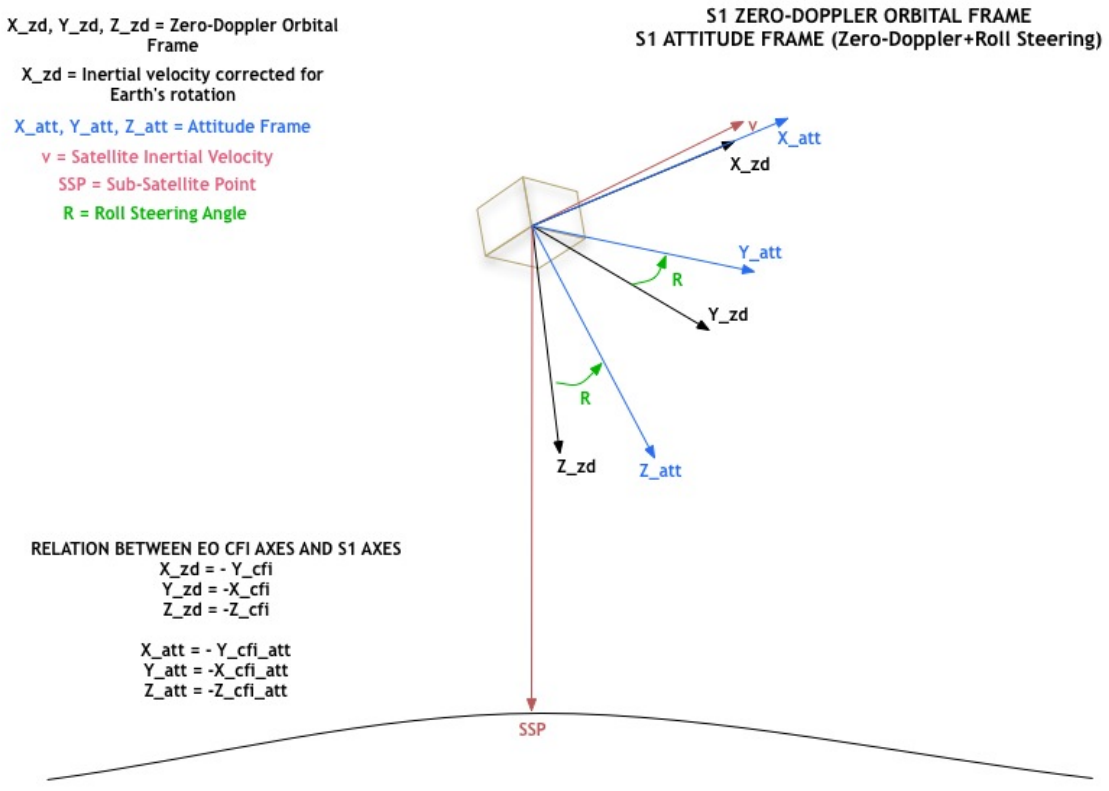
6.6 Correspondence Between EO CFI and Sentinel-1 Conventions

The following correspondences can be established between the EO CFI and the Sentinel-1 conventions:

a) Attitude Frames
The relation between the axes definition of the Sentinel-1 Satellite Attitude Frame and the EO CFI Satellite Attitude Frame is:

$$+\vec{X}_{SATATT}^{EOCFI} = -\vec{Y}_{SATATT}$$
$$+\vec{Y}_{SATATT}^{EOCFI} = -\vec{X}_{SATATT}$$
$$+\vec{Z}_{SATATT}^{EOCFI} = -\vec{Z}_{SATATT}$$


EO CFI NOMINAL ATTITUDE FRAME (Zero-Doppler)
EO CFI ATTITUDE FRAME ((Zero-Doppler+Roll Steering)
Z_cfi_zd
Z_cfi_att
X_cfi_zd
X_cfi_att
v
-Y_cfi_zd
-X_cfi_att
R
-X_cfi_zd
Y_cfi_xd
Y_cfi_att
R
-Z_cfi_zd
-Z_cfi_att
X_cfi_zd, Y_cfi_zd, Z_cfi_zd = Satellite Nominal Attitude Frame (EO CFI axes convention)
-Y_cfi_zd = Inertial velocity corrected for Earth's rotation
X_cfi_att, Y_cfi_att, Z_cfi_att = Attitude Frame (EO CFI axes convention)
R = Roll Steering Angle
v = Satellite Inertial Velocity
SSP = Sub-Satellite Point
SSP


X_zd, Y_zd, Z_zd = Zero-Doppler Orbital Frame
X_zd = Inertial velocity corrected for Earth's rotation
X_att, Y_att, Z_att = Attitude Frame
v = Satellite Inertial Velocity
SSP = Sub-Satellite Point
R = Roll Steering Angle
S1 ZERO-DOPPLER ORBITAL FRAME
S1 ATTITUDE FRAME (Zero-Doppler+Roll Steering)
v
X_att
X_zd
Y_att
R
Y_zd
R
Z_zd
Z_att
RELATION BETWEEN EO CFI AXES AND S1 AXES
X_zd = - Y_cfi
Y_zd = -X_cfi
Z_zd = -Z_cfi
X_att = - Y_cfi_att
Y_att = -X_cfi_att
Z_att = -Z_cfi_att
SSP

b) Quaternion Definition

The relation between the SAR Source Packet quaternion definition and the EO CFI quaternion definition is:

$$q_0^{EOCFI} = q_1$$
$$q_{11}^{EOCFI} = q_2$$
$$q_2^{EOCFI} = q_3$$
$$q_3^{EOCFI} = q_0$$

c) Quaternion ←→ Matrix

The following assumptions regarding the SAR Source Packet quaternions are applied:

- The equivalence between quaternion and matrix matches the one given in Section 6.3.
- The resulting matrix represents the transformation from Geocentric Mean of 2000 to Sentinel-1 Satellite Attitude Frame (see Section 6.5), so the matrix does not need to be transposed in order to generate the EOCFI quaternions