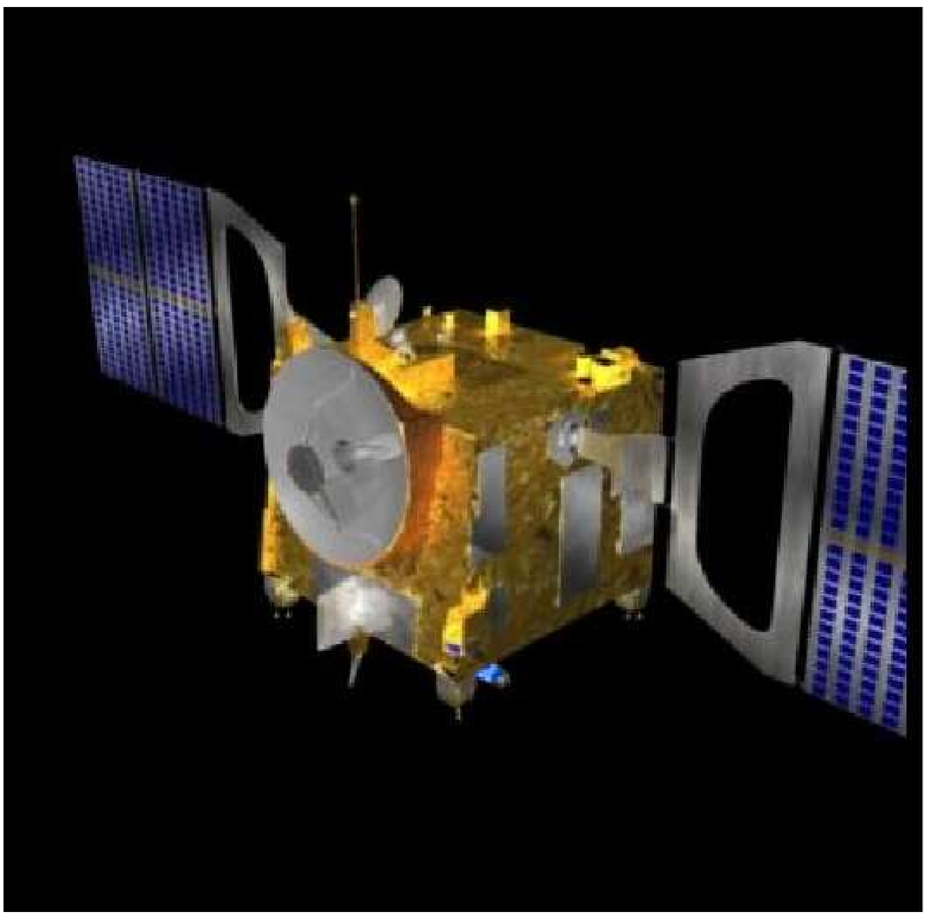
Aspera 4 : Frame definitions and sensors attitudes

PÈNE Nicolas
Centre d'Etude Spatiale des Rayonnements
Toulouse, France

21st December 2005



Contents

1 Introduction

The purpose of the document is to determine an algorithm of calculation of the *direction of the velocity vector (DVV)* of incidence particles.
Particle velocity vector is defined in the satellite frame of reference X_s, Y_s, Z_s.
Present document considers two separated packages : Mass spectrometer **IMA** and **Main Unit**.

Figure 1 shows the location of **IMA** and **Main Unit** with respect to the satellite frame.

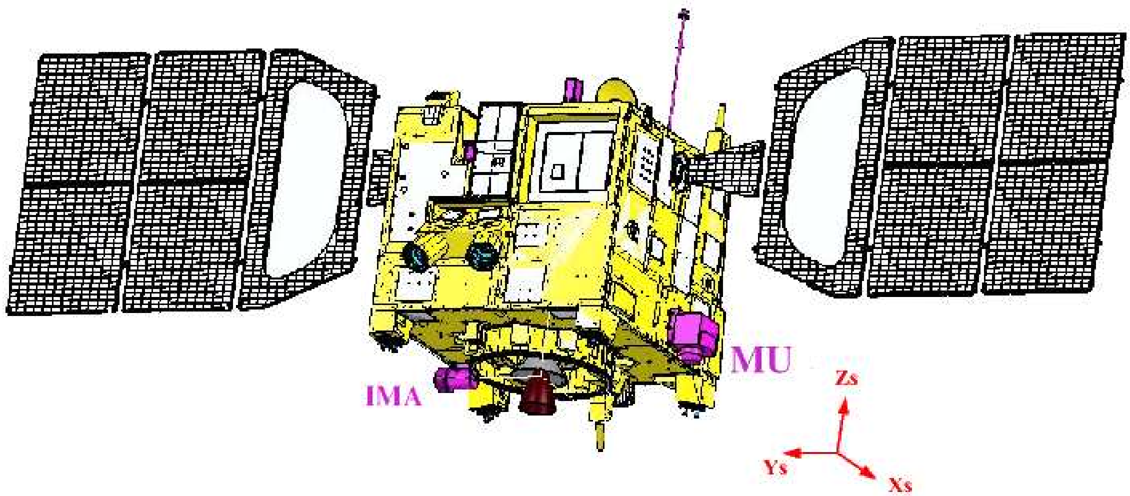


Figure 1: Satellite system ($X_sY_sZ_s$)

2 IMA

2.1 definition

Figures 2, 3, 4 and *5* shows location of IMA in the $X_sY_sZ_s$ VEX reference frame :

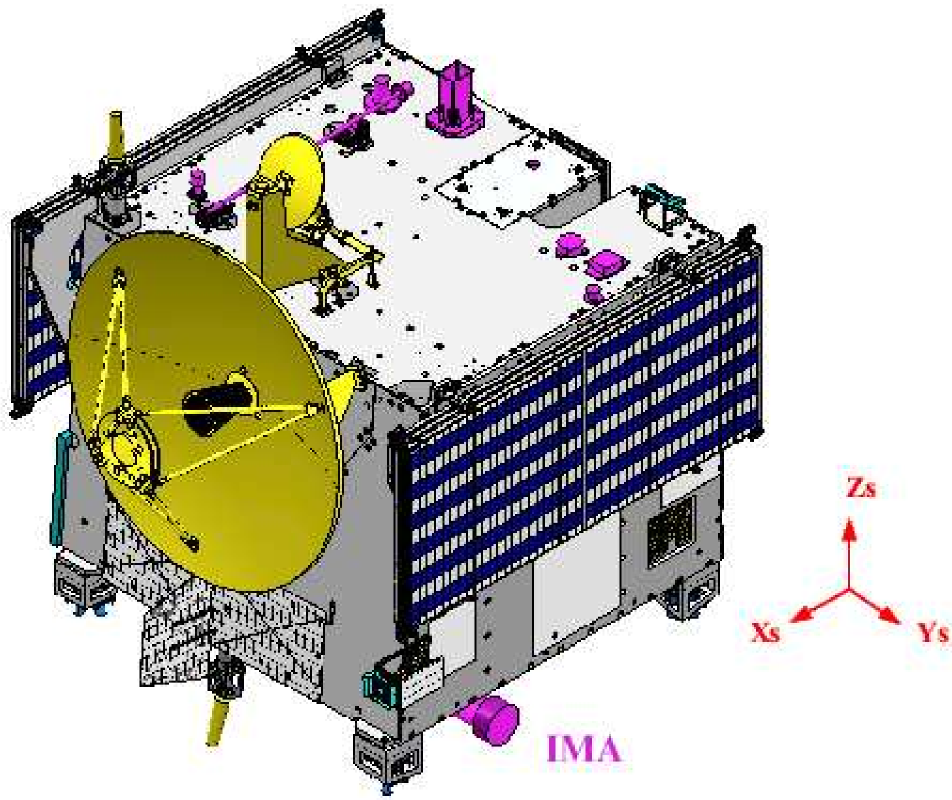


Figure 2: IMA location

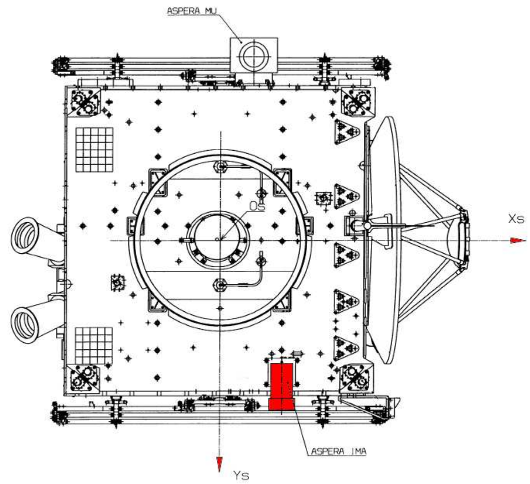


Figure 3: IMA location in $x_s y_s$ plane

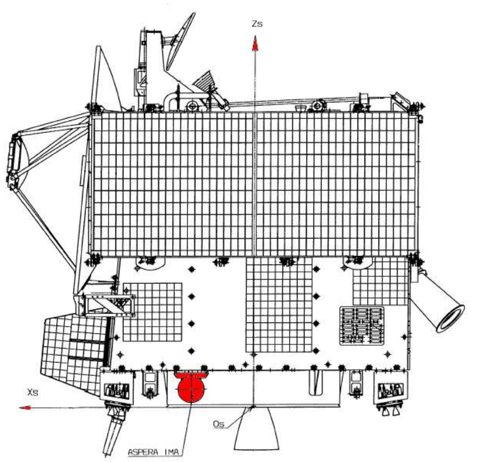


Figure 4: IMA location in $x_s z_s$ plane

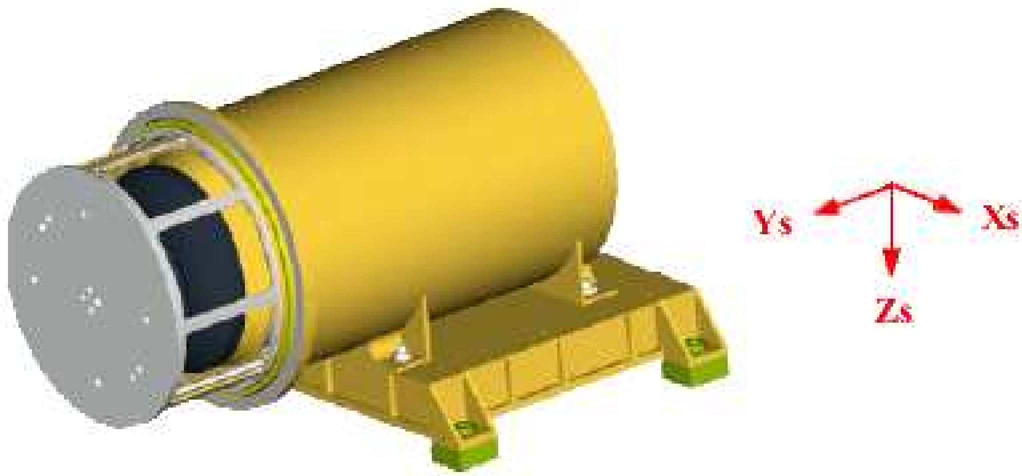


Figure 5: IMA orientation in $x_s y_s z_s$ plane

2.2 IMA variables

This section introduces the varibles used for IMA instrument.

θ is the **elevation angle** of the incident ion counted from the X_sZ_s plane towards $+Y_s$.
According to the calibration report, the maximum elevation angle is $\theta_{max} = 46\ degrees$, and the lowest value for the elevation angle is $\theta_{min} = -43.7\ degrees$.

ϕ is the **azimuthal angle** counted from the $+X_s$ axis towards the $+Z_s$ axis.

$\vec{u}$ is the DVV of the incident particle (unity vector) defined by θ and ϕ.

2.3 IMA $\vec{u}$ definition

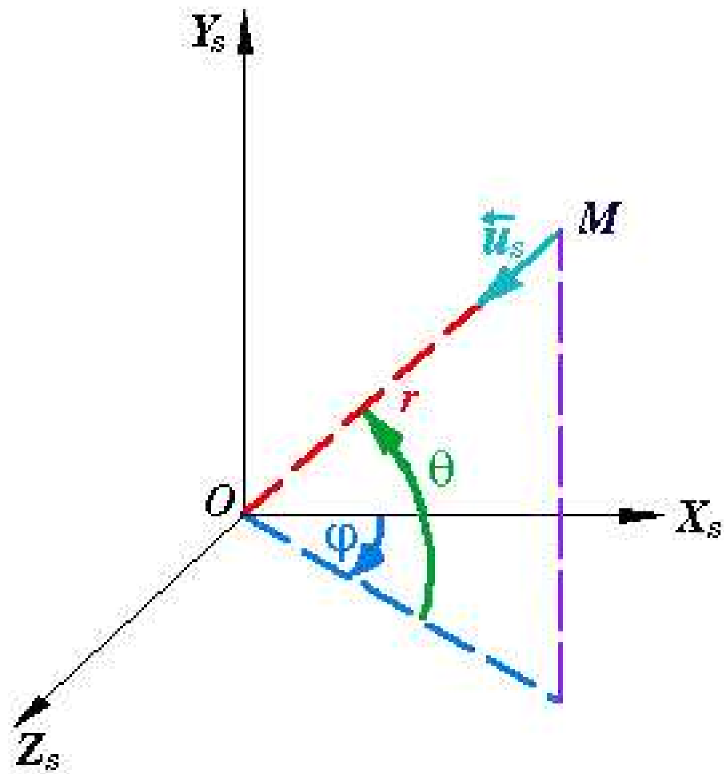


Figure 6: DVV in $X_sY_sZ_s$ plane

$$\begin{gathered}\theta : \text{ elevation angle} \\ \phi : \text{ azimuthal angle} \\ \overrightarrow{MO} : \text{ DVV}\end{gathered}$$

$\vec{u}$ follow the direction of $\overrightarrow{MO}$:

$$\vec{u} = \frac{\overrightarrow{MO}}{|MO|} = \frac{\overrightarrow{MO}}{r}$$

Then $\vec{u}$ components in the $X_s,\ Y_s,\ Z_s$ plane are:

$$\vec{u} = \left[\begin{array}{l} -\ cos\theta\ cos\phi \\ -\ sin\theta \\ -\ cos\theta\ sin\phi \end{array}\right.$$

2.4 IMA sector looking direction

Figure 7 shows the layout of 16 fields-of-view of azimuthal sectors of IMA. This view doesn't correspond to the physical sectors position.

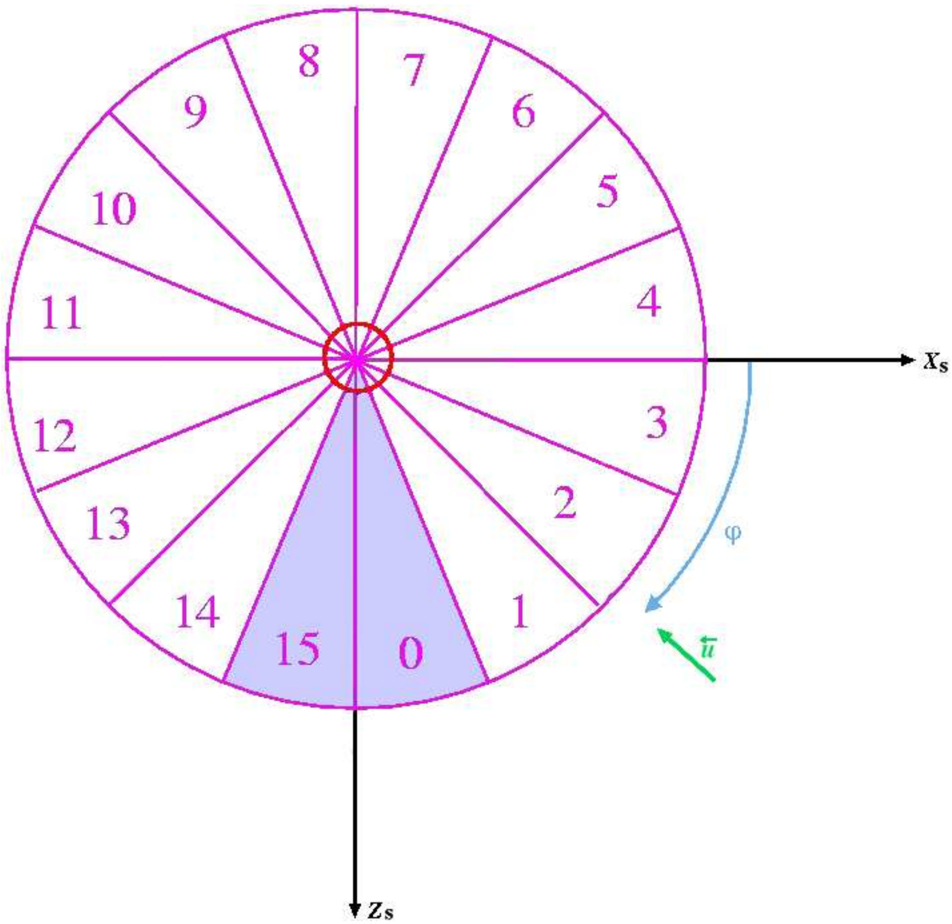


Figure 7: IMA sector looking direction in X_sZ_s plane

For example, an ion with $\vec{u}$ DVV shown in *figure 7* by green arrow will be registed in the azimuthal sector 2.

The following table displays the ϕ values at the beginning, the middle and the end of each sector.
It also indicates the value of $\vec{u}_{X_s Z_s}$ coming in the middle of each sector.

Sector	Begin	Middle	End	$\vec{u}_{X_s Z_s}$ (*)
0	90.00	78.75	67.50	(-0.195 , -0.981)
1	67.50	56.25	45.00	(-0.556 , -0.831)
2	45.00	33.75	22.50	(-0.831 , -0.556)
3	22.50	11.25	0.00	(-0.981 , -0.195)
4	0.00	348.75	337.50	(-0.981 , 0.195)
5	337.50	326.25	315.00	(-0.831 , 0.556)
6	315.00	303.75	292.50	(-0.556 , 0.831)
7	292.50	281.25	270.00	(-0.195 , 0.981)
8	270.00	258.75	247.50	(0.195 , 0.981)
9	247.50	236.25	225.00	(0.556 , 0.831)
10	225.00	213.75	202.50	(0.831 , 0.556)
11	202.50	191.25	180.00	(0.981 , 0.195)
12	180.00	168.75	157.50	(0.981 , -0.195)
13	157.50	146.25	135.00	(0.831 , -0.556)
14	135.00	123.75	112.50	(0.556 , -0.831)
15	112.50	101.25	90.00	(0.195 , -0.981)

3 Main Unit

3.1 Main Unit orientation

Main Unit (MU) is the common frame for four sensors : ELS, NPI, NPD1 and NPD2.

Main Unit (MU) can turn around Y_s axis.
To describe the instance attitudes of each sensor, we have to introduce a new refence frame $X_{mu}Y_{mu}Z_{mu}$ attached to MU.
We define $X_{mu}Y_{mu}Z_{mu}$ as a system equivalent to $X_sY_sZ_s$ when MU is in a *parking postion* shown in *figure 8*.

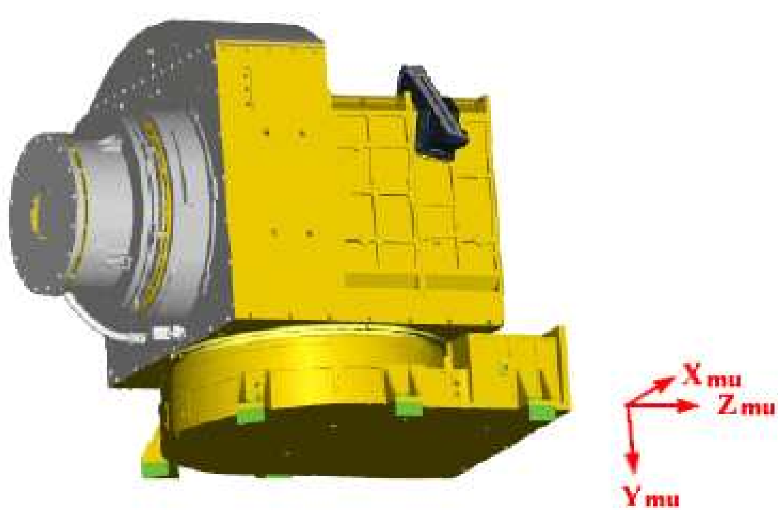


Figure 8: MU location

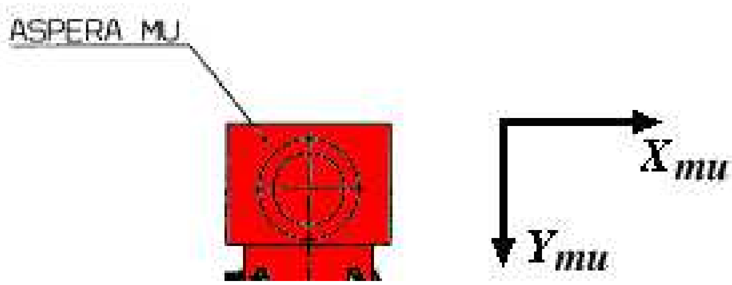


Figure 9: MU location in $x_{mu}y_{mu}$ plane

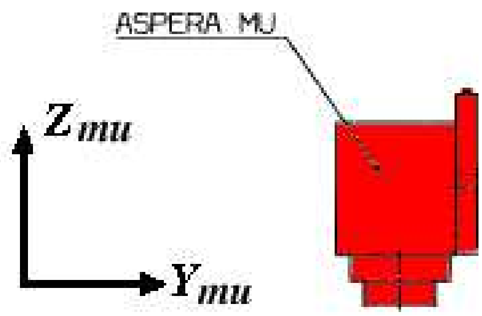


Figure 10: MU location in $y_{mu}z_{mu}$ plane

3.2 the spacecraft coordinates

Figure 12 shows MU position onto the spacecraft.
Here, main Unit is in a **parking position**.

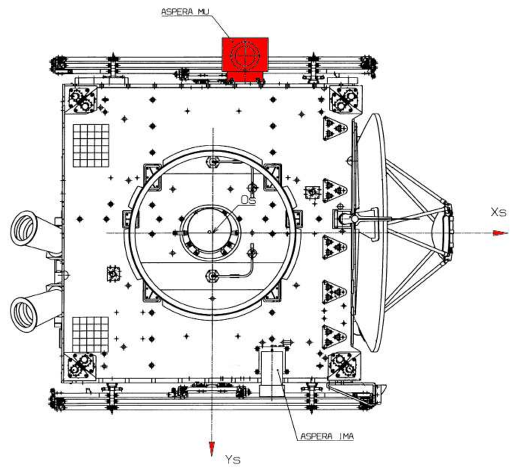


Figure 11: MU parking position in $x_s y_s$ plane

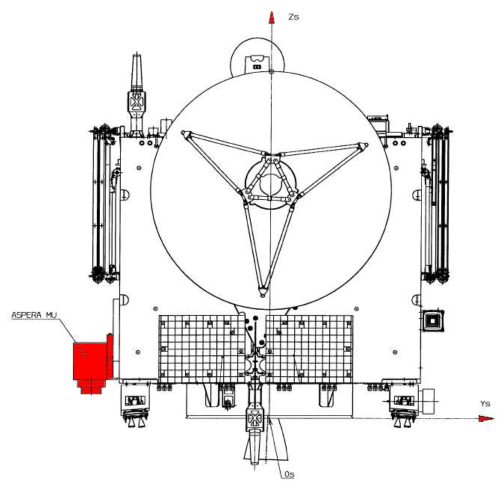


Figure 12: MU parking position in $y_s z_s$ plane

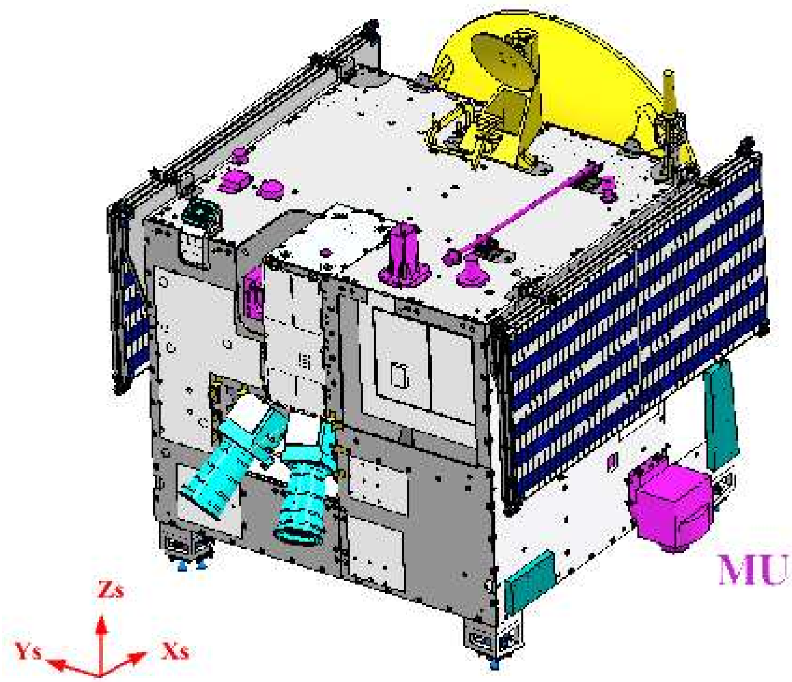


Figure 13: MU parking position in $x_{mu}y_{mu}z_{mu}$ plane

3.3 Main Unit variables

The following variables are used to determine DVV of particles measured by MU.

$Y_{mu} = Y_s$

θ is the **scanner angle** in the X_sZ_s plane of the satelite frame.
We define θ as the angle between Z_s and X_{mu}.
When MU turns around Y_s, θ is vareing from 0 to 180 degrees.

At parking position $\theta = 90$.
Then, for $\theta = 90$ degrees :

$$X_{mu} = X_s$$

$$Y_{mu} = Y_s$$

$$Z_{mu} = Z_s$$

Figure 14 represents X_{mu} and Y_{mu} in the X_sZ_s plane:

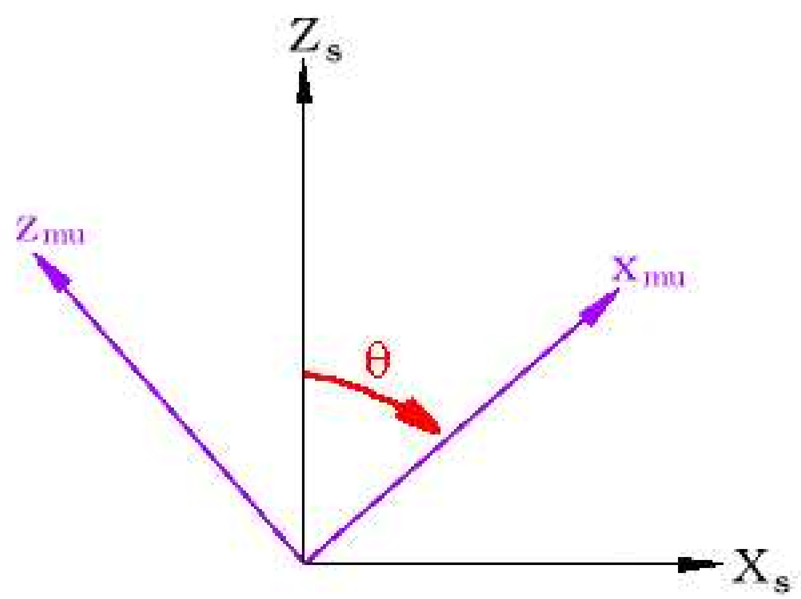


Figure 14: X_{mu} and Z_{mu} in the X_sZ_s plane

3.4 Rotation matrix

In $X_s Z_s$ system, X_{mu} and Z_{mu} are defined by:

$$X_{mu} = cos\theta \; Z_s \; + \; sin\theta \; X_s$$

$$Z_{mu} = sin\theta \; Z_s \; - \; cos\theta \; X_s$$

$$Y_{mu} = Y_s$$

so:

$$\begin{bmatrix} X_s \\ Y_s \\ Z_s \end{bmatrix} = \begin{bmatrix} sin\theta & 0 & -\cos\theta \\ 0 & 1 & 0 \\ cos\theta & 0 & sin\theta \end{bmatrix} . \begin{bmatrix} X_{mu} \\ Y_{mu} \\ Z_{mu} \end{bmatrix}$$

The rotation matrix to translate from the MU system to the satellite system is :

$$M = \begin{bmatrix} sin\theta & 0 & -\cos\theta \\ 0 & 1 & 0 \\ cos\theta & 0 & sin\theta \end{bmatrix}$$

3.5 *DVV* definition in the Main Unit system

ϕ is the **azimuthal angle** counted from the $+X_{mu}$ axis towards the $+Y_{mu}$ axis.

$\vec{u}_{mu}$ is an unity vector corresponding of DVV of the incident particle.

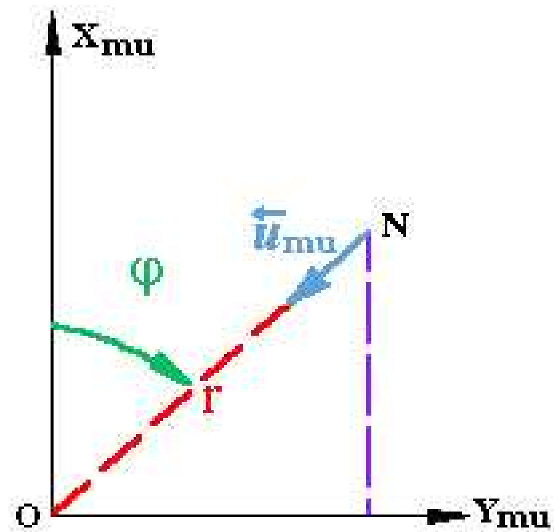


Figure 15: DVV in $X_{mu}Y_{mu}$ plane

$\vec{u}_{mu}$ follow the direction of $\overrightarrow{NO}$:

$$\vec{u}_{mu} = \frac{\overrightarrow{NO}}{|NO|} = \frac{\overrightarrow{NO}}{r}$$

note, that $u_{z_{mu}} = 0$

Thus, $\vec{u}_{mu}$ coordinates in the $X_{mu},\ Y_{mu},\ Z_{mu}$ plane are:

$$\vec{u}_{mu} = \begin{bmatrix} -\ cos\phi \\ -\ sin\phi \\ 0 \end{bmatrix}$$

3.6 *DVV* definition in the satellite system

$\vec{u}_s$ defines the particle direction in the satellite system.

To obtain $\vec{u}_s$ we have to apply the rotation matrix M to convert the $\vec{u}_{mu}$ from the MU system to the satellite system:

$$\begin{bmatrix} u_{x_s} \\ u_{y_s} \\ u_{z_s} \end{bmatrix} = \begin{bmatrix} sin\theta & 0 & -\cos\theta \\ 0 & 1 & 0 \\ cos\theta & 0 & sin\theta \end{bmatrix} . \begin{bmatrix} u_{x_{mu}} \\ u_{y_{mu}} \\ u_{z_{mu}} \end{bmatrix}$$

$$\begin{bmatrix} u_{x_s} \\ u_{y_s} \\ u_{z_s} \end{bmatrix} = \begin{bmatrix} sin\theta & 0 & -\cos\theta \\ 0 & 1 & 0 \\ cos\theta & 0 & sin\theta \end{bmatrix} . \begin{bmatrix} -\ cos\phi \\ -\ sin\phi \\ 0 \end{bmatrix}$$

Thus :

$$\vec{u}_s = \begin{bmatrix} -\ cos\phi\ sin\theta \\ -\ sin\phi \\ -\ cos\phi\ cos\theta \end{bmatrix}$$

Figure 16 shows an interpretation of the $\vec{u}_s$ in the X_s, Y_s, Z_s frame, where $\vec{u}_s$ is equal to $\frac{\overrightarrow{MO}}{|MO|}$:

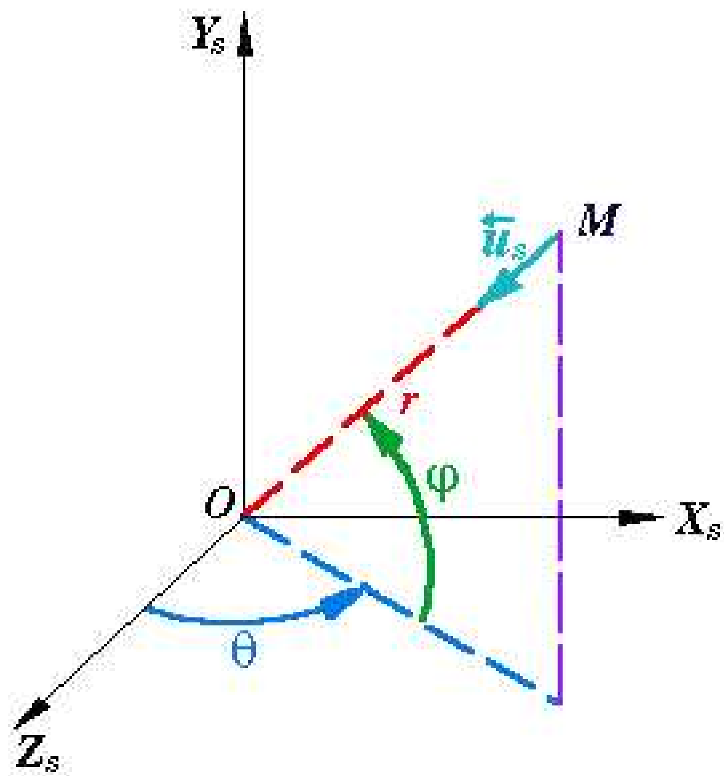


Figure 16: particle direction in $X_sY_sZ_s$ plane

θ : scanner angle
ϕ : azimuthal angle
$\overrightarrow{MO}$: DVV

3.7 ELS sector looking direction

The next picture defines fields-of-view of 16 ELS sectors in the X_{mu}, Y_{mu} frame.

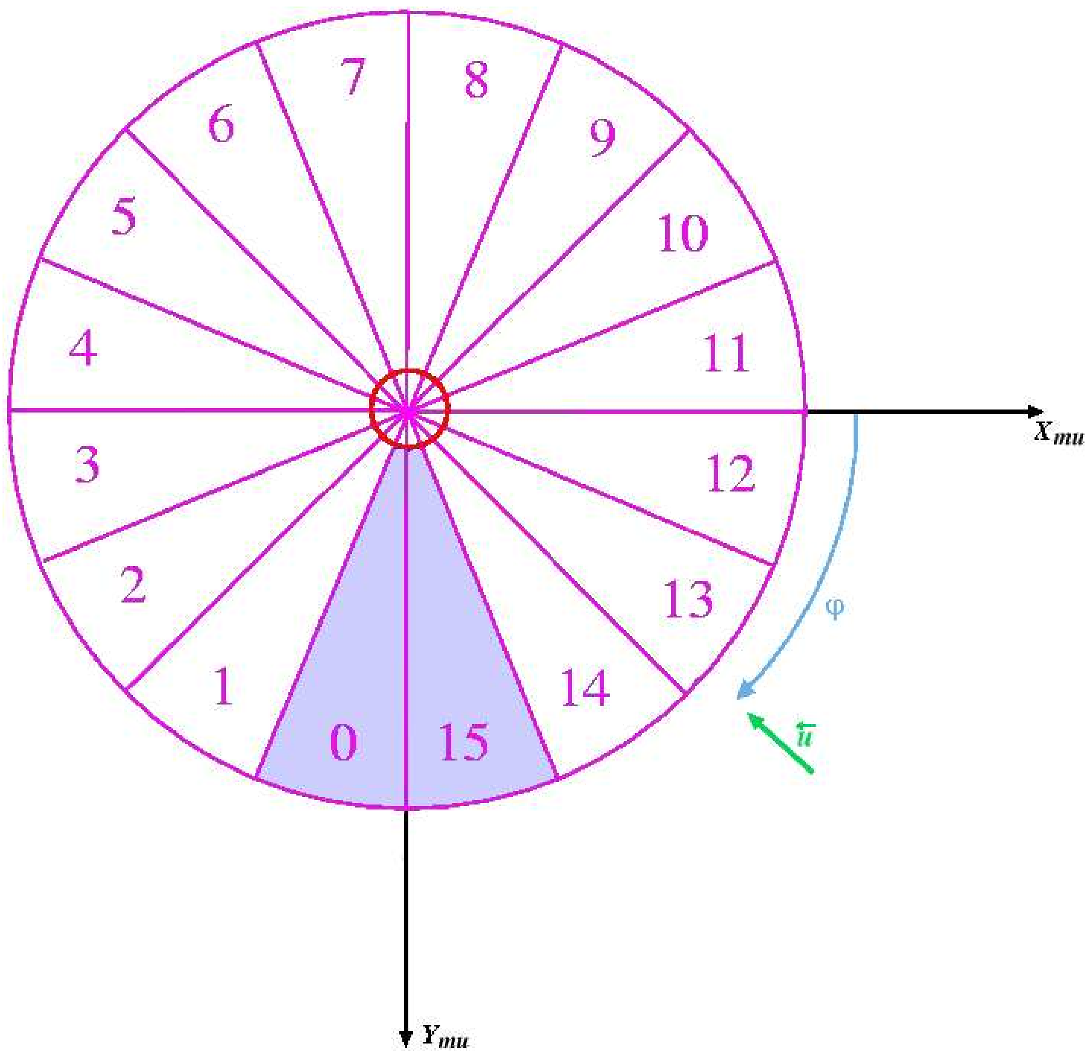


Figure 17: ELS sector looking direction in $X_{mu}Y_{mu}$ plane

ϕ : azimuthal angle
$\vec{u}$: unity vector corresponding to DVV

For example, a particle with $\vec{u}$ DVV shown in *figure 17* by green arrow will be registed by sector 13.

The following table displays the ϕ values at the beginning, the middle and the end of each sector.
It also indicates the value of $\vec{u}_{X_{mu}Y_{mu}}$ coming in the middle of each sector.

Sector	Begin	Middle	End	$\vec{u}_{X_{mu}Y_{mu}}$ *(*)*
0	90.00	101.25	112.50	(0.195 , -0.981)
1	112.50	123.75	135.00	(0.556 , -0.831)
2	135.00	146.25	157.50	(0.831 , -0.556)
3	157.50	168.75	180.00	(0.981 , -0.195)
4	180.00	191.25	202.50	(0.981 , 0.195)
5	202.50	213.75	225.00	(0.831 , 0.556)
6	225.00	236.25	247.50	(0.556 , 0.831)
7	247.50	258.75	270.00	(0.195 , 0.981)
8	270.00	281.25	292.50	(-0.195 , 0.981)
9	292.50	303.75	315.00	(-0.556 , 0.831)
10	315.00	326.25	337.50	(-0.831 , 0.556)
11	337.50	348.75	0.00	(-0.981 , 0.195)
12	0.00	11.25	22.50	(-0.981 , -0.195)
13	22.50	33.75	45.00	(-0.831 , -0.556)
14	45.00	56.25	67.50	(-0.556 , -0.831)
15	67.50	78.75	90.00	(-0.195 , -0.981)

3.8 NPI sector looking position

The next picture defines fields-of-view of 32 sectors of NPI in the X_{mu}, Y_{mu} frame.

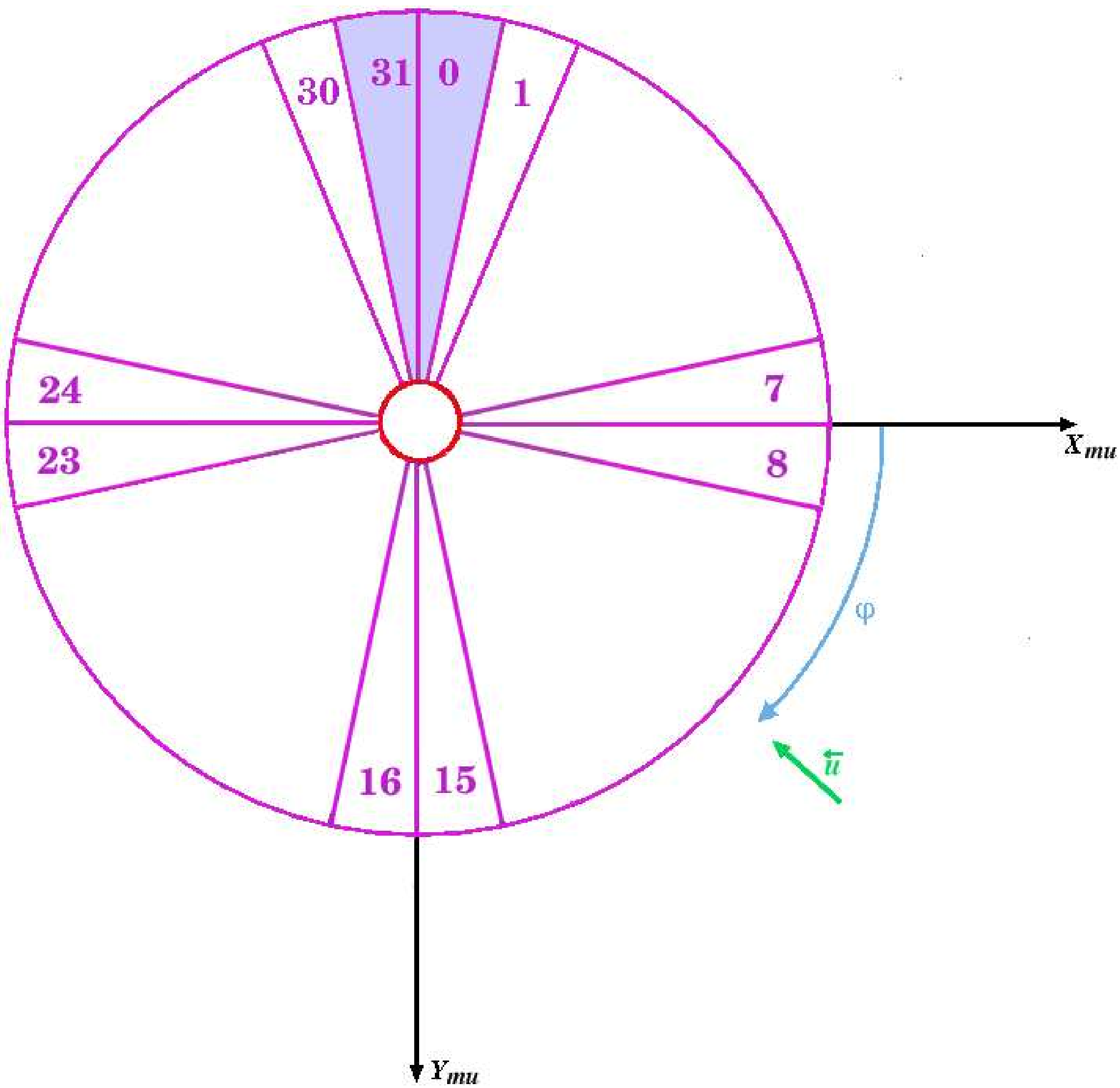


Figure 18: NPI sector looking direction in $X_{mu}Y_{mu}$ plane

ϕ : azimuthal angle
$\vec{u}$: unity vector corresponding to DVV.

The following table displays the ϕ values at the beginning, the middle and the end of each sector.
It also indicates the value of $\vec{u}_{X_{mu}Y_{mu}}$ coming in the middle of each sector.

Sector	Begin	Middle	End	$\vec{u}_{X_{mu}Y_{mu}}$ (*)
0	270.00	275.62	281.25	(-0.098 , 0.995)
1	281.25	286.88	292.50	(-0.290 , 0.957)
2	292.50	298.12	303.75	(-0.471 , 0.882)
3	303.75	309.38	315.00	(-0.634 , 0.773)
4	315.00	320.62	326.25	(-0.773 , 0.634)
5	326.25	331.88	337.50	(-0.882 , 0.471)
6	337.50	343.12	348.75	(-0.957 , 0.290)
7	348.75	354.38	0.00	(-0.995 , 0.098)
8	0.00	5.62	11.25	(-0.995 , -0.098)
9	11.25	16.88	22.50	(-0.957 , -0.290)
10	22.50	28.12	33.75	(-0.882 , -0.471)
11	33.75	39.38	45.00	(-0.773 , -0.634)
12	45.00	50.62	56.25	(-0.634 , -0.773)
13	56.25	61.88	67.50	(-0.471 , -0.882)
14	67.50	73.12	78.75	(-0.290 , -0.957)
15	78.75	84.38	90.00	(-0.098 , -0.995)
16	90.00	95.62	101.25	(0.098 , -0.995)
17	101.25	106.88	112.50	(0.290 , -0.957)
18	112.50	118.12	123.75	(0.471 , -0.882)
19	123.75	129.38	135.00	(0.634 , -0.773)
20	135.00	140.62	146.25	(0.773 , -0.634)
21	146.25	151.88	157.50	(0.882 , -0.471)
22	157.50	163.12	168.75	(0.957 , -0.290)
23	168.75	174.38	180.00	(0.995 , -0.098)
24	180.00	185.62	191.25	(0.995 , 0.098)
25	191.25	196.88	202.50	(0.957 , 0.290)
26	202.50	208.12	213.75	(0.882 , 0.471)
27	213.75	219.38	225.00	(0.773 , 0.634)
28	225.00	230.62	236.25	(0.634 , 0.773)
29	236.25	241.88	247.50	(0.471 , 0.882)
30	247.50	253.12	258.75	(0.290 , 0.957)
31	258.75	264.38	270.00	(0.098 , 0.995)

The sectors 14,15,16 and 17 are blocked by satellite body for any values of θ. Other sectors can be blocked for certain θ values.

3.9 NPD sector looking position

NPD consists of two sensors, NPD1 and NPD2.
Eanch sensor has 3 detectors in the azimutal plane.

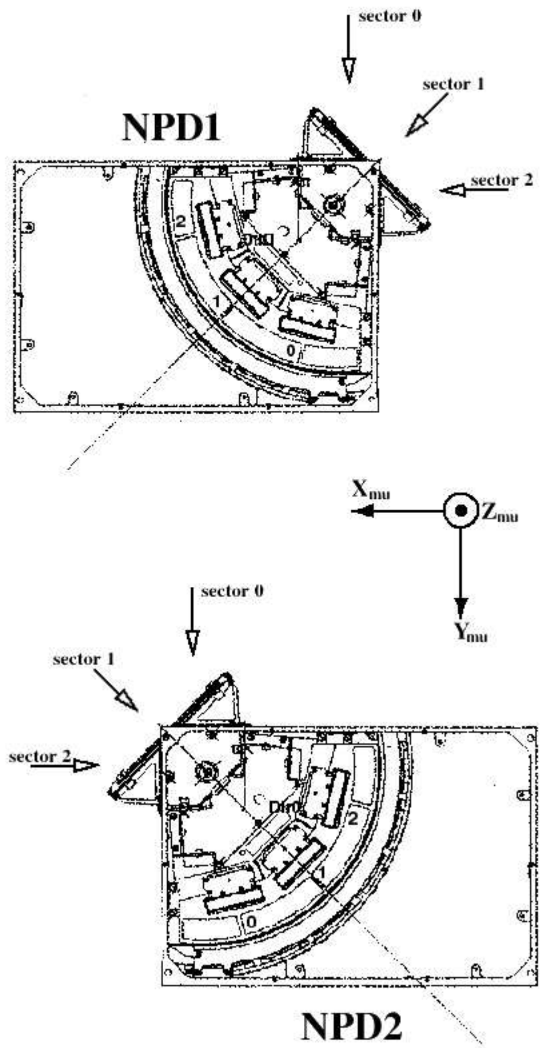


Figure 19: NPD sector looking direction in $X_{mu}Y_{mu}Z_{mu}$ plane

The following tables display for NPD1 and NPD2 the ϕ values at the beginning, the middle and the end of each sector.
It also indicates the value of $\vec{u}_{X_{mu}Y_{mu}}$ coming in the middle of each sector.

- NPD1 table :

Sector	Begin	Middle	End	$\vec{u}_{X_{mu}Y_{mu}}$ *(*)*
0	270.00	255.00	240.00	(0.259 , 0.966)
1	240.00	225.00	210.00	(0.707 , 0.707)
2	210.00	195.00	180.00	(0.966 , 0.259)

- NPD2 table :

Sector	Begin	Middle	End	$\vec{u}_{X_{mu}Y_{mu}}$ *(*)*
0	270.00	285.00	300.00	(-0.259 , 0.966)
1	300.00	315.00	330.00	(-0.707 , 0.707)
2	330.00	345.00	0.00	(-0.966 , 0.259)

Moreover the deflector is declined on 15° from rhe start surface plane :

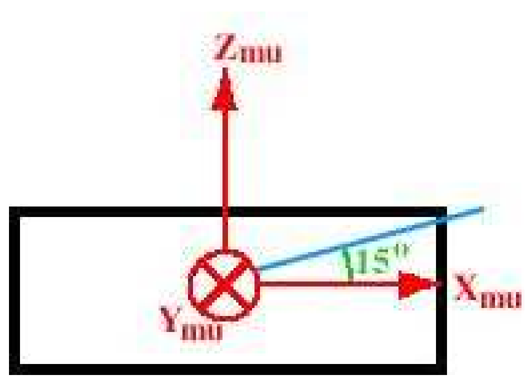


Figure 20: NPD deflection

3.10 Scanner datas

The scanner angle θ is determined in the telemetry files for each main unit sensor (ELS, NPI, NPD1 and NPD2).

In the header of this file, we find scanner direction, scanner speed and scanner position.

The following example shows how those information appear into the data file :

```
Scanner_Direction 1
Scanner_Speed 32
Scanner_Position 0
```

Telemetry frames are generated if $\theta \geq 0°$ and $\theta \leq 180°$. Nevertheless, the scanner can move between $-10°$ and $190°$.

position	CCW stop	CCW	CW	CW stop
deg count	$-10°$ 0	$0°$ 0	$180°$ 223	$190°$ ≤ 231

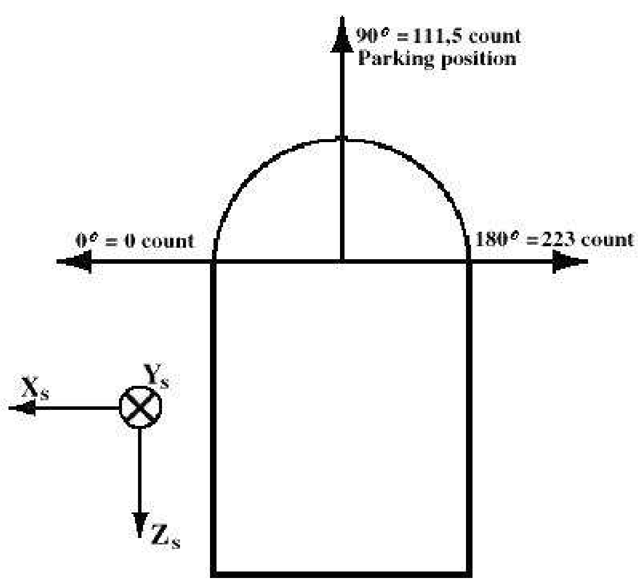


Figure 21: Scanner count

There is 3 allowed speeds for the scanner : 32 $sec/180°$, 64 $sec/180°$ and 128 $sec/180°$.

Scanner direction is defined as follows :

- *Scanner_direction 0 :* θ is counted up from 0 to 180 degrees.
- *Scanner_direction 1 :* θ is counted down from 180 to 0 degrees.

4 The VEXLIB

To switch to the VSO (Venus-Sun-Orbit) sytem, we have to use the **VEXLIB**. VEXLIB is developed by *GFI* (Daniel Popescu) and is using the SPICE LIB and OASW provided by *ESA*.

- *initSPICE(&ier, errtxt)* : initialisation of the SPICE lib

  - *errtxt* : error message if $ier! = 0$
  - *ier* : error code
    - 0 : Ok
    - 1 : error

- *initOrbJ2000(int *iunit, char *fname, int iorder,int *nvars, int *iframe, int *ibody ,int *itscal, double *tbeg, double *tend, int *ifid, int *ier)* : open an ESA binary orbit file created with *as2bin*

  - *iunit* : number of the unit associated to the orbit file.
  - *fname* : name of the orbit file.
  - *nvars* : number of vars in the orbit file.
    - 3 : position
    - 6 : position and velocity
    - 42 : position, velocity and derivative
  - *iframe* : default location.
    - 0 : mean equator and equinox of j2000
    - 1 : mean ecliptic and equinox of j2000
    - 2 : mean equator and equinox of b1950
    - 3 : mean ecliptic and equinox of b1950
  - *ibody* : system of reference.
    - 0 : bary-centre of the solar system
    - 2 : Venus
    - 3 : Earth
    - 11 : Sun
  - *itscal* : time ladder.
    - 0 : barycentric dynamic time (*MJD2000* format)
  - *tbeg* : beginning time of the orbit file.
  - *tbeg* : ending time of the orbit file.
  - *ifid* : orbit file id.
    - 0 : opening failed
  - *ier* : error code
    - 0 : Ok
    - 1 : unable to open file

2 : unable to get additional parameters

- *CalcStateJ2000(int ifid, double time, double *state, int *iframe, int *ibody, int *ier)* : calculate VEX state vector.

  - *ifid* : orbit file id.
  - *time* : time (barycentric dynamic time *MJD2000* format)
  - *state* : state vector in the *iframe* location and compared to *ibody*.
  - *iframe* : default location.
    0 : mean equator and equinox of j2000
    1 : mean ecliptic and equinox of j2000
    2 : mean equator and equinox of b1950
    3 : mean ecliptic and equinox of b1950
  - *ibody* : system of reference.
    0 : bary-centre of the solar system
    2 : Venus
    3 : Earth
    11 : Sun
  - *ier* : error code
    0 : Ok
    1 : time to early
    2 : time to late
    3 : time in a gap
    4 : error getting additional parameters
    5 : error can't read block header
    6 : invalid identifier

- *closeOrbJ2000(int ifid, int *ier)* : close an ESA binary orbit file.

  - *ifid* : orbit file id.
  - *ier* : error code
    0 : Ok
    1 : unable to close file
    2 : invalid identifier

- *initAttJ2000(int *iunit, char *fname, int iorder,int *nvars, int *iframe, int *itscal, double *tbeg, double *tend, int *ifid, int *ier)* : open an ESA binary attitude file.

  - *iunit* : number of the unit associated to the attitude file.
  - *fname* : name of the attitude file.
  - *iorder* : interpolation order (6 to 12; 8 recommended).
  - *nvars* : number of vars in the orbit file.
    3 : position
    6 : position and velocity
    42 : position, velocity and derivative
  - *ifname* : default location.
    Only *EMEJ2000* available in this version.
  - *itscal* : time ladder.
    0 : barycentric dynamic time (*MJD2000* format)
  - *tbeg* : beginning time of the attitude file.
  - *tbeg* : ending time of the attitude file.
  - *ifid* : attitude file id.
    0 : opening failed
  - *ier* : error code
    0 : Ok
    1 : unable to open file
    2 : unable to get additional parameters

- *CalcAttQuatJ2000(int ifid, double time, double *state, int *iframe, int *ibody, int *ier)* : calculate VEX quaternion attitude and angular velocity.

  - *ifid* : attitude file id.
  - *time* : time (barycentric dynamic time *MJD2000* format)
  - *state* : state vector in the *iframe* location and compared to *ibody*.
  - *iframe* : attitude reference.
    0 : J2000
  - *ibody* : system of reference.
    0 : bary-centre of the solar system
    2 : Venus
    3 : Earth
    11 : Sun
  - *ier* : error code
    0 : Ok
    1 : time to early
    2 : time to late
    3 : time in a gap

4 : error getting additional parameters
5 : error can't read block header
6 : invalid identifier

- *CalcAttMatJ2000(int ifid, double time, double amat[3][3], double omega[3], int *iframe, int *ier)*: calculate the angular velocity and the rotation matrix form the satellite location to the EMEJ2000 location.

  - *ifid* : attitude file id.
  - *time* : time (barycentric dynamic time *MJD2000* format)
  - *iframe* : attitude reference.
    0 : J2000
  - *ibody* : system of reference.
    0 : bary-centre of the solar system
    2 : Venus
    3 : Earth
    11 : Sun
  - *ier* : error code
    0 : Ok
    1 : time to early
    2 : time to late
    3 : time in a gap
    4 : error getting additional parameters
    5 : error can't read block header
    6 : invalid identifier

- *closeAttJ2000(int ifid, int *ier)* : close an ESA binary attitude file.

  - *ifid* : attitude file id.
  - *ier* : error code
    0 : Ok
    1 : unable to close file
    2 : invalid identifier

- *CalcJD2000(double *mjday, int year, int month, int day, int hour, int minute, double second*): convert a calendar date into a julian day.

  - *mjday* : Julian day modified.

- *CalcDJ2000(double mjday, int *year, int *month, int *day, int *hour, int *minute, double *second*): convert a julian day into a calendar date.

  - *mjday* : Julian day to modify.

- *eme2000vso(double date, double pmat[3][3], double pmatt[3][3], int* ier, char *errtxt*): changeover matrix of the EME2000 towards VSO.

  - *date* : barycentric dynamic time (*MJD2000* format)
  - *pmat* : changeover matrix $(3 * 3)$ EME2000 to VSO
  - *pmatt* : transposed matrix of *pmat*
  - *ier* : error code
    0 : Ok
    1 : error
  - *errtxt* : error message if $ier != 0$

- *eme2000uai(double date, double pmat[3][3], double pmatt[3][3], int* ier, char *errtxt*): changeover matrix of the EME2000 towards UAI.

  - *date* : barycentric dynamic time (*MJD2000* format)
  - *pmat* : changeover matrix $(3 * 3)$ EME2000 to Venus fixed
  - *pmatt* : transposed matrix of *pmat*
  - *ier* : error code
    0 : Ok
    1 : error
  - *errtxt* : error message if $ier != 0$

- *eme2000gse(double date, double pmat[3][3], double pmatt[3][3], int* ier, char *errtxt*): changeover matrix of the EME2000 towards GSE.

  - *date* : barycentric dynamic time (*MJD2000* format)
  - *pmat* : changeover matrix $(3 * 3)$ EME2000 to GSE
  - *pmatt* : transposed matrix of *pmat*
  - *ier* : error code

0 : Ok
1 : error
- *errtxt* : error message if $ier! = 0$

- *MxV(double matrix[3][3], double vin[3], double vout[3])*
  : Multiplication of a vector (3) by a matrix (3x3).

  - *matrix* : Matrix $(3 * 3)$
  - *vin* : incoming vector (3)
  - *vout* : result of the multiplication; vector (3).

- *calcStateGSE(double date, int target, double *state, int* ier, char *errtxt)*
  : position and velocity of an object in GSE.

  - *date* : barycentric dynamic time (*MJD2000* format)
  - *target* : id NAIF of the object
  - *state* : state vector in the *iframe* location and compared
  - *ier* : error code
    0 : Ok
    1 : error
  - *errtxt* : error message if $ier! = 0$