Small Bodies Ineteroperability

PDAP for Hayabusa

ISAS/JAXA

Iku Shinohara

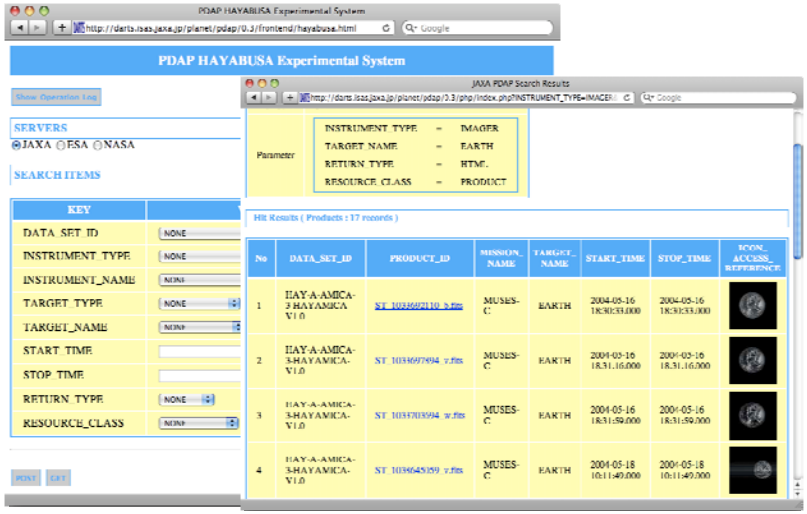
Yukio Yamamoto

Background

- For the small bodies interoperability, there were two big problems:
  - A small body is not always regular shape !
    - Indeed, asteroid Itokawa shows its irregular shape.
    - Currently, this problem is not resolved enough, but brings up some matter.
  - There is no appropriate resource defined by PDAP version 0.3.
    - The IMAGE_PRODUCT assumes that all images have four definable coordinates on four corners.
    - New RESOURCE: FLYBY_PRODUCT was defined, and works well in the current implementation.

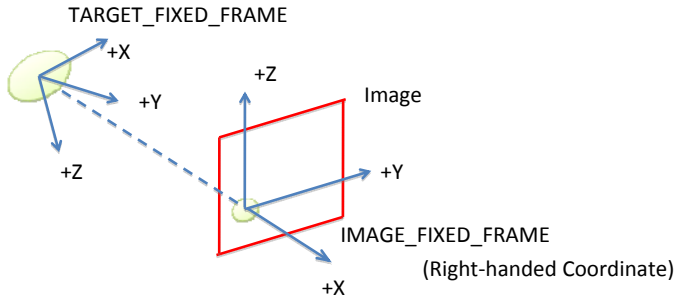
HAYABUSA PDAP

http://darts.isas.jaxa.jp/planet/pdap/index.html



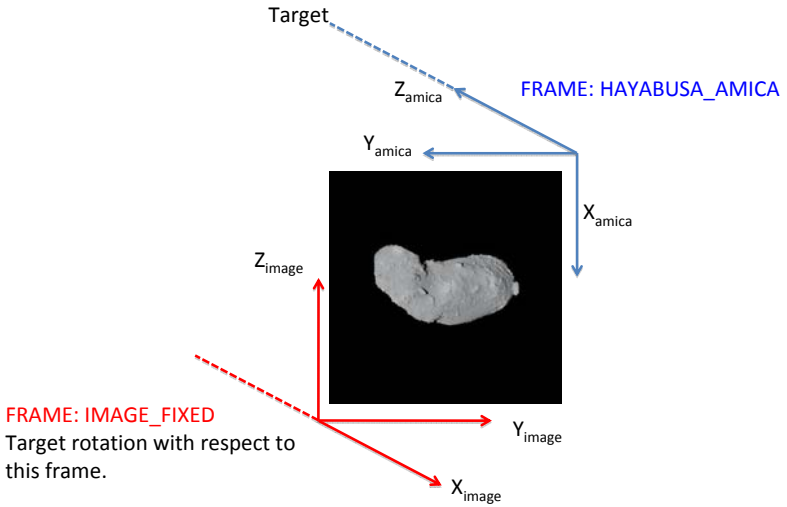
Target Roll, Pitch, Yaw angle

Target Roll, Pitch, and Yaw angles are defined referencing IMAGE_FIXED_FRAME.



Hayabusa FlybyProduct

Definition of IMAGE_FIXED_FRAME



Definition of Target Rotation order from IMAGE_FIXED to TARGET_FIXED

Definition: $Roll:\theta_x, Pitch:\theta_y, Yaw:\theta_z$

Frame Rotation Order: $\theta_z \Rightarrow \theta_y \Rightarrow \theta_x$

Coordinate transformation:

$\vec{r}_{\text{image}}$: a position relative to the frame IMAGE_FIXED

$\vec{r}_{\text{target}}$: the same position relative to the frame TARGET_FIXED

$$\vec{r}_{\text{target}} = (R_x \cdot R_y \cdot R_z) \cdot \vec{r}_{\text{image}}$$

$$\vec{r}_{\text{image}} = (R_z^{-1} \cdot R_y^{-1} \cdot R_x^{-1}) \cdot \vec{r}_{\text{target}}$$

$$R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(-\theta_x) & -\sin(-\theta_x) \\ 0 & \sin(-\theta_x) & \cos(-\theta_x) \end{pmatrix} \quad R_y = \begin{pmatrix} \cos(-\theta_y) & 0 & \sin(-\theta_y) \\ 0 & 1 & 0 \\ -\sin(-\theta_y) & 0 & \cos(-\theta_y) \end{pmatrix} \quad R_z = \begin{pmatrix} \cos(-\theta_z) & -\sin(-\theta_z) & 0 \\ \sin(-\theta_z) & \cos(-\theta_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Hayabusa FlybyProduct

Calculation using SPICE

AMICA DATA

Program with SPICE TOOLKIT

```
pxform_c("IMAGE_FIXED", "ITOKAWA_FIXED", et, mat);
m2eul_c(mat, 1, 2, 3, &roll, &pitch, &yaw);
roll *= -dpr_c(); pitch *= -dpr_c(); yaw *= -dpr_c();
```

Relational Database (Additional Columns of FlybyProducts)

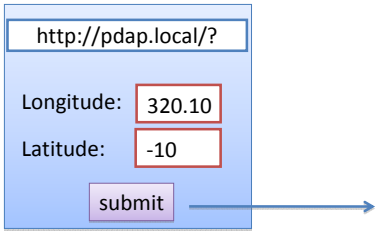
Distance [km]	Phase Angle [deg]	Roll [deg]	Pitch[deg]	Yaw [deg]
14.63073	8.5087	177.9672	8.5087	-117.4137
…				

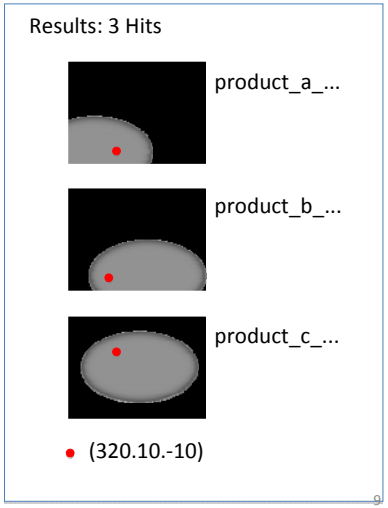
Problem of Hayabusa PDAP

- The definition of PDAP is not unique because Euler angles are not unique to express even for the same rotation. Quaternion is an expression to express a rotation uniquely, but it is not intuitively recognizable expression.
- Euler angles themselves are not so friendly. For instance, user would like to find out an image with longitude and latitude, but flyby product cannot respond for this request directory

PDAP Hayabusa

Use-Case Idea





The most useable input is pair of (longitude, latitude) , but…

This interface will make the PDAP System complicated.

FlybyProduct Processing

User Input

(λ,β)

$\lambda :=$ longitude

$\beta :=$ latitude

PDAP Database

Set of $(r,\alpha,\theta_x,\theta_y,\theta_z)$

$r :=$ Distance To Target

$\alpha :=$ Target Phase Angle

$\theta_x :=$ Target Roll Angle

$\theta_y :=$ Target Pitch Angle

$\theta_z :=$ Target Yaw Angle

Process

$$\vec{u} = \begin{pmatrix} \cos\beta \cdot \cos\lambda \\ \cos\beta \cdot \sin\lambda \\ \sin\beta \end{pmatrix}$$

$$R_x^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{pmatrix}$$

$$R_y^{-1} = \begin{pmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{pmatrix}$$

$$R_z^{-1} = \begin{pmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\vec{v} = (R_z^{-1} \cdot R_y^{-1} \cdot R_x^{-1}) \cdot \vec{u}$$

Judge

$\vec{v} = (v_x, v_y, v_z)$

If $v_x \geq 0$ then

user input point may be visible.

PDAP System may not respond quickly for all flybyproducts !