# Derived Quantities

## March 2006

**Overheard in a corridor during a coffee break…**

***"Cartesian state vectors and quaternions are nice, but I'm going to hold out for a latitude and longitude, and maybe even a phase angle."***

## Overview



- **What are "derived quantities?"**
- **A quick tour of some of the routines provided for the computation of derived quantities**
  - **Vector/Matrix Routines**
  - **Geometry Routines**
  - **Coordinate System Routines**
- **Computing Illumination Angles**
- **Computing Ring Plane Intercepts**
- **Computing Occultation Events**

# What are Derived Quantities?

Navigation and Ancillary Information Facility

- Derived quantities are data produced from states, C-matrices, frame transformations, physical constants, time conversions, etc.
- Some examples are:
  - Angles, Angular Rates
  - Distances, Speeds
  - Directions
  - Lighting conditions
  - Cartographic parameters
  - Schedules of events
  - etc.
- The SPICE Toolkit contains many routines that assist with the computations of derived quantities.
  - Some are fairly low level, some are quite high level.
  - More are being added as time permits.

- Derived quantities are "the holy grail" of SPICE!



# A Quick Tour

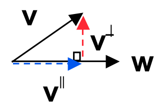
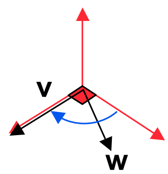
Navigation and Ancillary Information Facility

- Vector/Matrix Routines
  - Vector and vector derivative arithmetic
  - Matrix arithmetic
- Geometric "Objects"
  - Planes
  - Ellipses
  - Ellipsoids
  - Rays
- Coordinate Systems
  - Spherical: latitude/longitude, co-latitude/longitude, right ascension/declination; Geodetic, Cylindrical, Rectangular, Planetographic
- Others

The lists on the following pages are just a subset of what's available in the Toolkit.

# Vectors

Navigation and Ancillary Information Facility

| Function | Routine |
|---|---|
| <v,w> | VDOT, DVDOT |
| **v** x **w** | VCROSS, DVCRSS |
| **v**/\|**v**\| | VHAT, DVHAT |
| **v** x **w** / \| **v** x **w**\| | UCROSS, DUCRSS |
| **v** + **w** | VADD, VADDG |
| **v** - **w** | VSUB, VSUBG |
| a**v** [+ b**w** [+ c**u**]] | VSCL, [VLCOM, [VLCOM3]] |
| angle between **v** and **w** | VSEP |
| \|**v**\| | VNORM |
|  | VPROJ, VPERP |
|  | TWOVEC, FRAME |





# Matrices

Navigation and Ancillary Information Facility

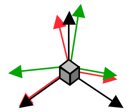
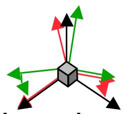
## Selected Matrix-Vector Linear Algebra Routines

| Function | Routine |
|---|---|
| M x v | MXV |
| M x M | MXM |
| $M^t$ x v | MTXV |
| $M^t$ x M | MTXM |
| M x $M^t$ | MXMT |
| $v^t$ x M x v | VTMV |
| $M^t$ | XPOSE |
| $M^{-1}$ | INVERSE, INVSTM |

M = Matrix
V = Vector
X = Multiplication
T = Transpose

# Matrix Conversions

Navigation and Ancillary Information Facility

| Function | | | Routines |
|---|---|---|---|
|  Euler angles | ← Transform between → | $\begin{matrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{matrix}$ 3x3 rotation matrix | – `EUL2M, M2EUL` |
|  Euler angles and Euler angle rates *or* rotation matrix and angular velocity vector | ← Transform between → | $\begin{matrix} a_x & a_y & a_z & & 0 & \\ b_x & b_y & b_z & & & \\ c_x & c_y & c_z & & & \\ \alpha_x & \alpha_y & \alpha_z & a_x & a_y & a_z \\ \beta_x & \beta_y & \beta_z & b_x & b_y & b_z \\ \gamma_x & \gamma_y & \gamma_z & c_x & c_y & c_z \end{matrix}$ 6x6 state transformation matrix | – `EUL2XF, XF2EUL` `RAV2XF, XF2RAV` |
|  Rotation axis and angle | ← Transform between → | $\begin{matrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{matrix}$ 3x3 rotation matrix | – `RAXISA, AXISAR` `ROTATE, ROTMAT` |
| $(Q_0, Q_1, Q_2, Q_3)$ Quaternion | ← Transform between → | $\begin{matrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{matrix}$ 3x3 rotation matrix | – `Q2M, M2Q` |



# Geometry

Navigation and Ancillary Information Facility

| Function | Routine |
|---|---|
| • **Ellipsoids** | |
| – nearest point [derivative] | – `NEARPT, SUBPT, DNEARP` |
| – surface ray intercept | – `SURFPT, SRFXPT` |
| – surface normal | – `SURFNM` |
| – limb | – `EDLIMB` |
| – slice with a plane | – `INELPL` |
| – altitude of ray w.r.t. to ellipsoid | – `NPEDLN` |
| • **Planes** | |
| – intersect ray and plane | – `INRYPL` |
| • **Ellipses** | |
| – project onto a plane | – `PJELPL` |
| – find semi-axes of an ellipse | – `SAELGV` |

# Position Coordinate Transformations

Navigation and Ancillary Information Facility

| Coordinate Transformation | Routine |
|---|---|
| – Latitudinal to/from Rectangular | – LATREC<br>RECLAT |
| – Planetographic to/from Rectangular | – PGRREC<br>RECPGR |
| – R.A. Dec to/from Rectangular | – RADREC<br>RECRAD |
| – Geodetic to/from Rectangular | – GEOREC<br>RECGEO |
| – Cylindrical to/from Rectangular | – CYLREC<br>RECCYL |
| – Spherical to/from Rectangular | – SPHREC<br>RECSPH |



# Velocity Coordinate Transformations - 1

Navigation and Ancillary Information Facility

| • Coordinate Transformation | • Jacobian (Derivative) Matrix Routine |
|---|---|
| – Latitudinal to/from Rectangular | – DRDLAT<br>DLATDR |
| – Planetographic to/from Rectangular | – DRDPGR<br>DPGRDR |
| – R.A. Dec to/from Rectangular | – DRDLAT*<br>DLATDR* |
| – Geodetic to/from Rectangular | – DRDGEO<br>DGEODR |
| – Cylindrical to/from Rectangular | – DRDCYL<br>DCYLDR |
| – Spherical to/from Rectangular | – DRDSPH<br>DSPHDR |

* Jacobian matrices for the R.A and Dec to/from rectangular mappings are identical to those for the latitudinal to/from rectangular mappings

# Velocity Coordinate Transformations - 2



- **Transform velocities from one coordinate system to another using the SPICE Jacobian matrix routines. Example:**
  - Let ( x, y, z ) be a time-dependent vector expressed in rectangular coordinates:

    $( x, y, z ) = \Gamma(t)$
  - Let ( $\alpha, \beta, \gamma$ ) be the same vector expressed in spherical coordinates:

    $( \alpha, \beta, \gamma ) = \Phi ( \Gamma(t) )$
  - Then the chain rule gives us the time derivative of the position in spherical coordinates:

    $d (\Phi( \Gamma(t) ) ) / dt \;=\; J (\Gamma(t)) * d \Gamma(t) / dt$
  - The left hand side above is the velocity in spherical coordinates. The first factor on the right is the "Jacobian matrix" of the mapping from rectangular to spherical coordinates; the second factor on the right is the velocity in rectangular coordinates.



NAIF

# Velocity Coordinate Transformations - 3

Navigation and Ancillary Information Facility

- **The SPICE calls that implement this computation are:**

```
CALL SPKEZR ( TARG, ET, REF, CORR, OBS, STATE, LT )
CALL DSPHDR ( STATE(1), STATE(2), STATE(3), JACOBI )
CALL MXV    ( JACOBI,   STATE(4), SPHVEL )
```

- **After these calls, the vector SPHVEL contains the velocity in spherical coordinates: specifically, the derivatives**

  ( d (r) / dt, d (colatitude) / dt, d (longitude) /dt )
- **Caution: coordinate transformations often have singularities, so derivatives may not exist everywhere.**
  - Exceptions are described in the headers of the SPICE Jacobian matrix routines.
  - SPICE Jacobian matrix routines signal errors if asked to perform an invalid computation.

# Other Derived Quantities

Navigation and Ancillary Information Facility

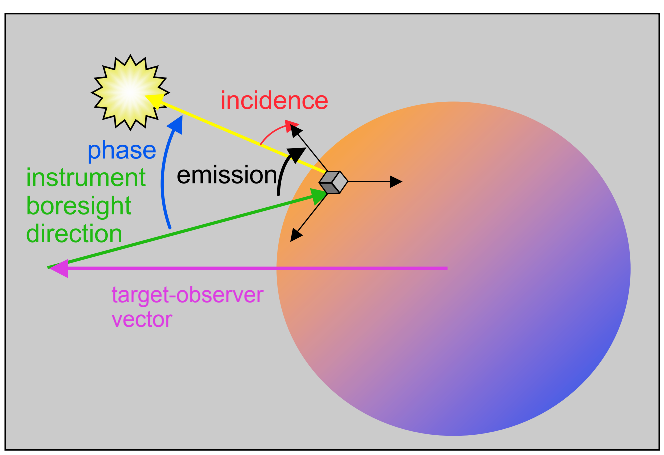
- Illumination angles (phase, incidence, emission)
  - ILLUM
- Subsolar point
  - SUBSOL
- Subobserver point
  - SUBPT
- Surface intercept point
  - SRFXPT
- Longitude of the sun ($L_s$), an indicator of season
  - LSPCN



# Computing Illumination Angles

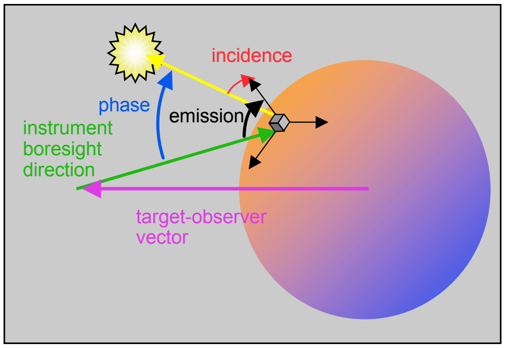
Navigation and Ancillary Information Facility

- Given the direction of an instrument boresight in a bodyfixed frame, return the illumination angles (incidence, phase, emission) at the surface intercept on a tri-axial ellipsoid



# Computing Illumination Angles

Navigation and Ancillary Information Facility



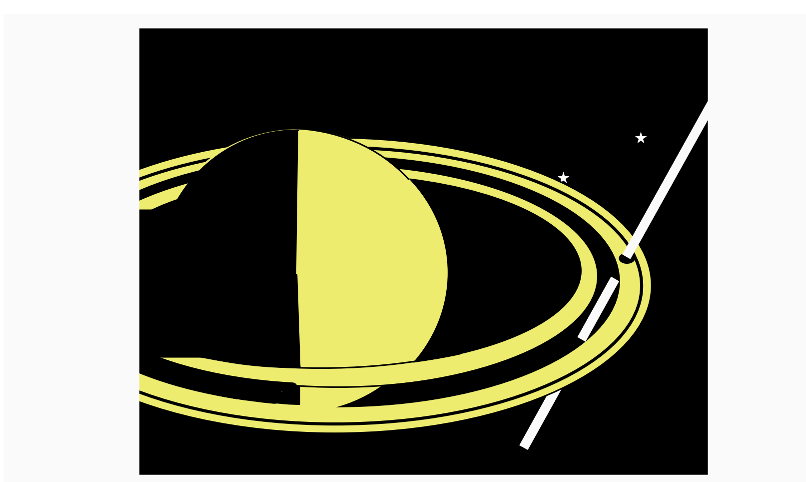

- CALL GETFOV to obtain boresight direction vector.
- CALL SRFXPT to find intersection of direction vector with surface.
- CALL ILLUM to determine illumination angles.



# Computing Ring Plane Intercepts

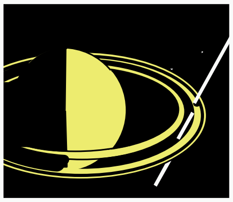
Navigation and Ancillary Information Facility

- Determine the intersection of the apparent line of sight vector between Earth and Cassini with Saturn's ring plane and determine the distance of this point from the center of Saturn.



# Computing Ring Plane Intercepts

Navigation and Ancillary Information Facility



This simplified computation ignores the difference between the light time from Saturn to the observer and the light time from the intercept point to the observer. It also ignores the difference in stellar aberration between the observer-Saturn vector and the observer-intercept vector.

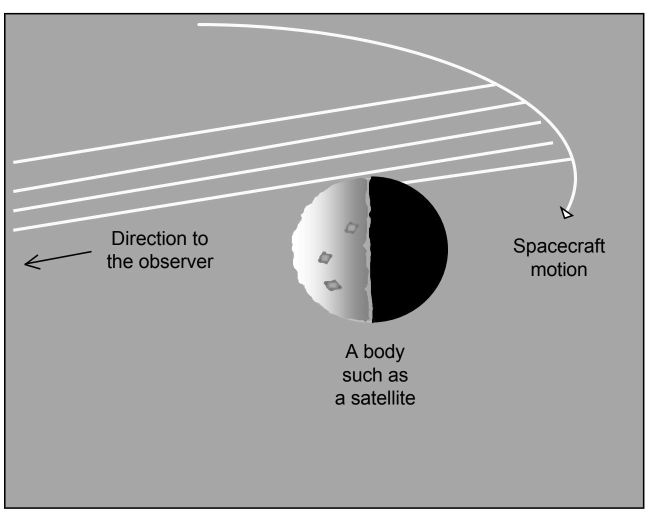
- CALL SPKEZR to get apparent position of spacecraft as seen from earth at time ET in J2000 reference frame SCVEC.
- CALL SPKEZR to get apparent position of center of Saturn at time ET as seen from earth in J2000 frame SATCTR.
- CALL PXFORM to get rotation from body-fixed coordinates to J2000 at light-time corrected epoch. The third column of this matrix gives the pole direction of Saturn in the J2000 frame SATPOL.
- CALL NVP2PL and use SATCTR and SATPOL to construct the ring plane RPLANE.
- CALL INRYPL to intersect Earth-spacecraft vector SCVEC with the Saturn ring plane RPLANE to produce the intercept point X.
- CALL VSUB to get position of intercept point with respect to the Saturn XSAT (subtract SATCTR from X) and use VNORM to get the distance of XSAT from the center of Saturn.



# Computing Occultation Events

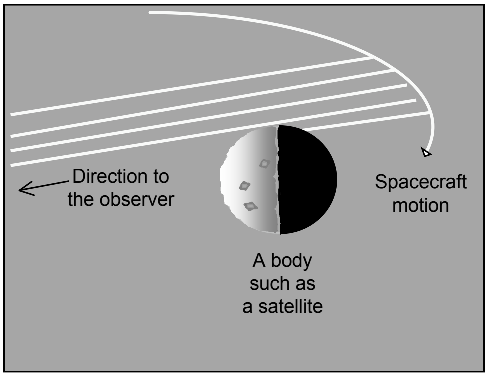
Navigation and Ancillary Information Facility

- Determine when the spacecraft will be occulted by an object (such as a natural satellite) as seen from an observer (such as earth).



# Find Epoch of Occultation Ingress

Navigation and Ancillary Information Facility




- **Select a start epoch, stop epoch and step size.**
  - Start and stop epochs should bracket a single occultation ingress epoch.
  - Step size should be smaller than occultation duration, but large enough to solve problem with reasonable speed.
- **WHILE ( ET < "stop epoch" )**
  - CALL SRFXPT to see if the apparent direction of the spacecraft as seen by the observer intersects the surface of the body; do this by examining the "FOUND" flag. Take note of any such transition.
  - If an intersection is found, check that the spacecraft is further from the observer than the intersection point.
    - » Can now refine time of occultation ingress via binary search.
    - » Exit loop.
  - Otherwise, increment ET.



# Event Finding Software

Navigation and Ancillary Information Facility

- **NAIF is busy building new software that will find a large number of geometric events, such as occultations, transits, periapsis passage, etc.**
- **A preview of these new capabilities will be provided in a later presentation.**