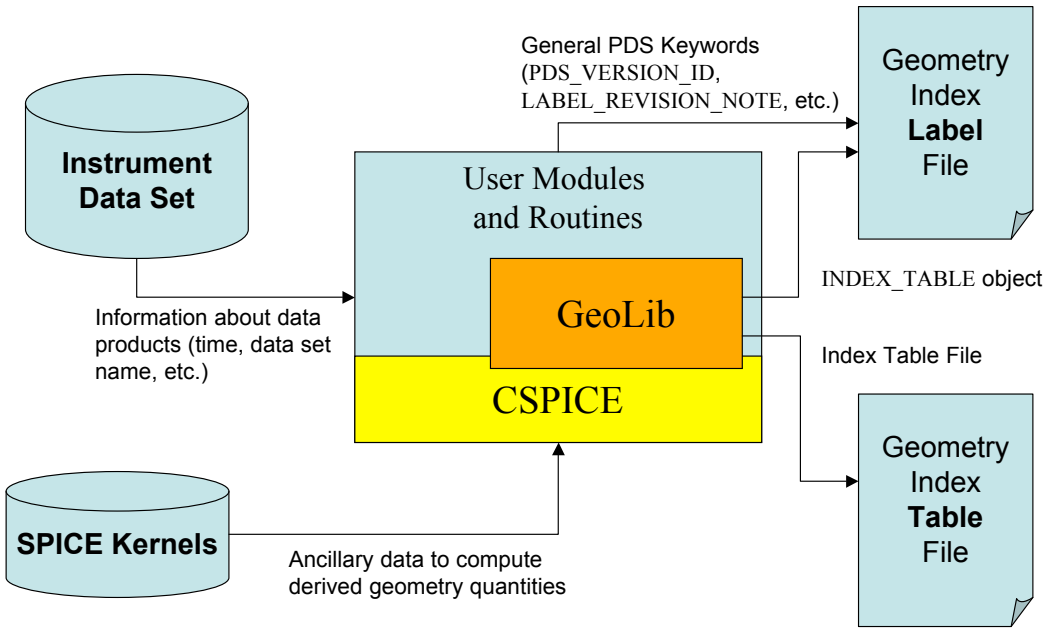
Geometry Library: Using GeoLib

April 2005

Overview

- Usage
- Typical Algorithm
- Examples:
  - For Mapping Instruments
  - For Non-mapping Instruments

GeoLib Usage: Geometry Index Generation



Typical Algorithm

- For each data product, retrieve information about the measurements:
  - Start and End Time
  - Modes of Operation
- Generate the number of samples required:
  - Lines (start and end points) for mapping instruments
  - Time for non-mapping instruments
- Generate one index line (row) per sample:
  - Generic Parameters
  - Instrument Dependent Parameters (only for mapping instruments)

Example: Non-Mapping Instrument (I)

```
/*  This example shows how to compute, for a
    single product, the required lines of the
    geometry index file */
#include "SpiceUsr.h"
#include "geolib.h"
    .
/*  Define the required variables */
FILE           *F;
FILE           *LBL;
SpiceChar      targ[20];
SpiceDouble    time;

SpiceDouble    ALT;
SpiceDouble    SDIST;
SpiceDouble    SLON;
SpiceDouble    LTST;
SpiceDouble    SSLAT;
SpiceDouble    SSLON;
SpiceDouble    SCLAT;
SpiceDouble    SCLON;
SpiceDouble    SSPOS[3];
SpiceDouble    SSVEL[3];
SpiceDouble    STPOS[3];
SpiceDouble    STVEL[3];
SpiceInt       ORBN;

SpiceChar      CARRAY[5];
SpiceInt       IARRAY[5];
SpiceDouble    DARRAY[36];
```

```
int            nlines;
int            rlen;
int            clen[46];

/* Load the SPICE kernels */
furnsh_c("geolib.ker");

/* Open Geometry Index table file for writing
   */
F = fopen( "MY_GEOIDX.TAB", "w+" );

/*  Get Observation time and all other
    relevant information (user defined module).
    This information is obtained form the data
    product label files:
    - Start and End times
    - Release/revision information
    - File/Path Name
    - Data Set/Product ID
    - Target */

/*  Get the times of all the required samples
    (user defined module) */
```

Example: Non-Mapping Instrument (II)

```
/*  For each of the samples obtained in the
    previous step, compute all the required
    parameters, by calling the following
    routines */
FOR sample[0] to sample[n-1] {

/* Compute all non-instrument related
    parameters, based on times */
orbit_number( time, &ORBN );
solar_longitude( time, targ, &SLON);
sub_solar_point( time, targ, &SSLON, &SSLAT);
sc_sun_distance( time, &SDIST );
sc_sun_state_vector( time, SSPOS, SSVEL );
sc_target_state_vector( time, targ, STPOS,
    STVEL);
spacecraft_altitude( time, targ, &ALT);
sub_spacecraft_point( time, targ, &SCLON,
    &SCLAT);
local_true_solar_time( time, targ, SCLON,
    &LTSL);

/*  Assign the rest of the keywords to N/A and
    generate arrays required by writegln (user
    defined module):
    CARRAY: all character keywords;
    IARRAY: all integer keywords;
    DARRAY: all float keywords */
/* Write Index Line */
writegln( F, CARRAY, IARRAY, DARRAY );

} /* END of FOR loop */
```

```
/*  Once processed all samples, write the
    Index Label file */

/*  Remove blank columns, get number of lines
    of the index file, length of record and
    length of data columns */
clnblank( F, &nlines, &rlen, clen);

/*  Open the Geometry Index label file for
    writting */
LBL = fopen( "MY_GEOIDX.LBL", "w");

/*  Write the general PDS keywords and assign
    them the appropriate values (User defined
    module) */

/*  Write the INDEX_TABLE object in the label
    file */
wrtidxobj( LBL, nlines, rlen, clen);

/*  Close both index and label files */
fclose( F );
fclose( LBL );

/* Unload the kernel pool */
unload_c("geolib.ker");
```

Example: Mapping Instrument (I)

```
/*  This example shows how to compute, for a
    single product, the required lines of the
    geometry index file */
#include "SpiceUsr.h"
#include "geolib.h"
    .
/*  Define the required variables */
FILE          *F;
FILE          *LBL;
SpiceChar     targ[20];
SpiceDouble   time;

SpiceDouble   ALT;
SpiceDouble   SDIST;
SpiceDouble   SLON;
SpiceDouble   LTST;
SpiceDouble   SSLAT;
SpiceDouble   SSLON;
SpiceDouble   SCLAT;
SpiceDouble   SCLON;
SpiceDouble   SSPOS[3];
SpiceDouble   SSVEL[3];
SpiceDouble   STPOS[3];
SpiceDouble   STVEL[3];
SpiceDouble   LPT[2], RPT[2];
SpiceDouble   CPT[2];
SpiceDouble   SLANT;
SpiceDouble   PHASE, SOLAR, EMISSN;
SpiceDouble   NPOLE, SUBSC, SUBSUN;
SpiceDouble   HPS, VPS;
```

```
SpiceInt      ORBN;

SpiceChar     CARRAY[5];
SpiceInt      IARRAY[5];
SpiceDouble   DARRAY[36];

int           nlines;
int           rlen;
int           clen[46];

/* Load the SPICE kernels */
furnsh_c("geolib.ker");

/* Open Geometry Index table file for writing
   */
F = fopen( "MY_GEOIDX.TAB", "w+" );

/*  Get Observation time and all other
    relevant information (user defined module).
    This information is obtained form the data
    product label files:
    - Start and End times
    - Release/revision information
    - File/Path Name
    - Data Set/Product ID
    - Target */

/*  Compute Footprint, corner points and time
    for each of the required lines (user
    defined module) */
```

Example: Mapping Instrument (II)

```
/*  For each of the 'lines' obtained in the
    previous step, compute all the required
    parameters, by calling the following
    routines */
FOR sample[0] to sample[n-1] {

/* Compute all non-instrument related
    parameters, based on times */
orbit_number( time, &ORBN );

solar_longitude( time, targ, &SLON);
sub_solar_point( time, targ, &SSLON, &SSLAT);
sc_sun_distance( time, &SDIST );
sc_sun_state_vector( time, SSPOS, SSVEL );
sc_target_state_vector( time, targ, STPOS,
    STVEL);
spacecraft_altitude( time, targ, &ALT);
sub_spacecraft_point( time, targ, &SCLON,
    &SCLAT);

/* Get lat/lon for central point of each line
    describing the footprint */
center_point( targ, time, LPT, RPT, CPT);

/* Compute the LTST: CPT[0] = LON */
local_true_solar_time( time, targ, CPT[0],
    &LTSL);

/* Compute Instrument Related Parameters */
illumination_angles( time, targ, CPT, &PHASE,
    &SOLAR, &EMISSION );
slant_distance( time, targ, CPT, &SLANT);
```

```
/*  The following parameters required the use
    of the IK; this file has to be load
    beforehand. */
azimuth_angles( targ, instr, CPT, time, &NPOLE,
    &SUBSC, &SUBSUN);
pixel_scale( time, targ, instr, &HPS, &VPS);

/*  Generate arrays required by writegln (user
    defined module):
    CARRAY: all character keywords;
    IARRAY: all integer keywords;
    DARRAY: all float keywords */

/* Write Index Line */
writegln( F, CARRAY, IARRAY, DARRAY );

} /* END of FOR loop */
```

Example: Mapping Instrument (III)

```
/*  Once processed all samples, write the
    Index Label file */

/*  Remove blank columns, get number of lines
    of the index file, length of record and
    length of data columns */
clnblank( F, &nlines, &rlen, clen);

/*  Open the Geometry Index label file for
    writting */
LBL = fopen( "MY_GEOIDX.LBL", "w");

/*  Write the general PDS keywords and assign
    them the appropriate values (User defined
    module) */

/*  Write the INDEX_TABLE object in the label
    file */
wrtidxobj( LBL, nlines, rlen, clen);

/*  Close both index and label files */
fclose( F );
fclose( LBL );

/* Unload the kernel pool */
unload_c("geolib.ker");
```