# Mars Science Laboratory

## Software Interface Specification

## PLACES Data Products for PDS

## Release: 1.3

## Rev -A

**JPL D-78442**

**MSL-477-3984**

**2016-07-01**

Prepared by:

______________________

Robert Deen, MIPL/OPGS RDR Cognizant Engineer

Jet Propulsion Laboratory

California Institute of Technology

Pasadena, California

# 1. Introduction

This document describes the structure and use of data extracted from the PLACES (Position Localization and Attitude Correction Estimate Storage) database for the PDS (Planetary Data System) archive. These data cover primarily rover localizations and mechanism positions, along with a few ancillary items. The first few sections provide conceptual context; what is PLACES and how the data should be interpreted. The last sections contain the specific directory layouts and format specifications for the PDS delivery.

## *1.1. What is Localization*

Rover localization is the process of determining the position (or location) and orientation (or attitude) of the rover at any given time. The true value is unknowable; only estimates (called "solutions") of the correct value exist. These estimates will differ depending on how they were produced, and may sometimes conflict with each other. Yet each estimate is valid for its intended purpose by its intended audience.

Localization solutions are produced both onboard the rover, and by ground analysis of data.

## *1.2. Primary Purpose of PLACES*

The primary purpose of PLACES is to manage all these solutions, from users throughout the operations and science teams as well as those produced by the rover itself, and provide a coherent view of them across the Project.

Thus, PLACES is a "one-stop shopping" source for localizations. It applies no value judgments to the localizations; each is potentially as useful as any other. The user specifies which set of solutions to use (the View mechanism), and the database supplies those solutions in an easy-to-use manner.

## *1.3. PLACES database vs. PDS extraction*

PLACES is an active database, accessed via ReST-style web services. This means that each function is accessed via a URL and all queries and updates are atomic. All services of the database can be exercised using a simple Web browser, but are also available via command-line tools and programmatic requests. PLACES contains many active queries, for example performing on-the-fly coordinate frame transformations.

However, such a database is incompatible with the PDS archive. The purpose of PDS is to archive data in ways that will be accessible and usable decades if not centuries after the mission ends. It is unlikely that any active database will still be usable in that time frame, especially with no funding for maintenance. Therefore, for the PDS delivery, all the data have been extracted into static XML or text files that are easily interpretable without special tools. The preparers of the archive have attempted to provide the data in as useful a form as possible, in some cases running active queries on the database to extract the output files.

The intent of this document is not to describe the active database; only the format and use of the extracted files is explicitly described. However, references to the database are common during explanations and examples. In general, "PLACES" refers to the data, the database, or both, as should be clear from context.

For more information on the PLACES database itself, see [ 2 ] (which is an operations document, not part of the PDS archive).

## 1.4. High-Level Requirements

The requirements for the PLACES database are:

- Provide a single Project-wide source for localization solutions.
- Accept and store localization inputs from any team member.
- Provide access to all submitted raw localization data.
- Provide a simple and convenient mechanism to answer common queries.
- Provide views on the localization data that present a single position/orientation value for each Rover Motion Counter (RMC).
- Guarantee immutability of results, so the same query always returns the same answer (subject to some restrictions).
- Provide access via Web service (ReST-style queries), Web page, programmatic request, and summary XML files.
- Store mechanism (joint angle) information as extracted from telemetry.
- Store other related ancillary data.

# 2. Background

This section provides some background on the MSL mission that is useful for understanding PLACES.

## *2.1. Coordinate Frames and References*

A variety of coordinate frames are used on the MSL mission; there are literally dozens. Of these, however, only four are relevant to rover localizations.

Note that PLACES works entirely in an XYZ Cartesian coordinate system, even for orbital data. It does not keep track of nor use latitude and longitude, although some products present latitude and longitude for convenience (see Section 4.4.2.4).

### 2.1.1. Rover (Nav) Frame

The Rover Navigation Frame (usually called just Rover frame in this document) is attached to the rover, with the origin fixed at a point on the nominal ground, between the middle wheels. The +X axis points out the front of the rover, with +Z pointing away from the belly ("down"), and +Y completing the right-handed system. Thus any part of the rover that doesn't move has a fixed coordinate in the Rover frame. However, the entire frame "moves" relative to the surface during drives, and thus the numeric coordinates of rocks as measured in a Rover frame change.

Thus there exist "instances" of the Rover frame, indexed by the RMC (see Section 2.2). Each instance is valid during a certain period of time, during which the coordinates of surface objects in Rover frame do not change.

The origin and orientation of these Rover frame instances describe the rover's location throughout time. Refining knowledge of these origins is the primary goal of rover localization.

Note that there exists a Rover Mechanical frame, which is closely related to Rover Navigation but with a constant offset. It is not used in the PLACES system.

Because the Rover frame is attached to the rover, the position and attitude of the rover itself at any given time is the same as the position and attitude of the rover frame at the same time (assuming both are measured with respect to the same parent frame, which is normally Site). The concepts of rover pose and Rover Frame pose are thus synonymous, and the terms are used interchangeably in this document.

### 2.1.2. Site Frame

Because the coordinates of surface objects measured in Rover frame change whenever the rover drives, it is advantageous to have a frame that is fixed to the surface.

However, on-board rover position estimation is rather imprecise; errors of 10% are common, more if the wheels slip excessively. Onboard Visual Odometry (visodom) reduces this error significantly, but does not eliminate it, and is not always used due to time and power constraints. If only one surface-fixed frame were available, accumulation of error over the mission would be significant. Therefore multiple Site frames are used, each fixed to the surface at a certain place.

The Site frame thus defines the general vicinity of the rover. Most operations are carried out relative to the current Site frame. When the rover moves too far away from the current Site origin, a new Site is declared, with a new origin, and the Site index is incremented.

Sites allow isolation of localization errors. If it is determined that a particular drive was improperly localized early in the mission, only the origins of subsequent Site frames need to be corrected. All data within a subsequent Site are unaffected, because they are defined relative to their own Site origin.

Site frames are defined with +X pointing north, +Z pointing down according to the local gravity vector, and +Y completing the right-handed-system (east). The origin coincides with the Rover (Nav) frame at the time the site is declared.

### 2.1.3. Local Level Frame

Local Level frame is a combination of Site and Rover frames. It shares an origin with the Rover frame, thus moving with the rover and being instanced via the RMC. However, it shares the orientation of the Site frame, pointing north/down. The Local Level frame can always be easily computed given both the Site and Rover frames, therefore no explicit information about Local Level frames is stored in the database. However, Derived-from frames (see Section 3.7) are commonly expressed in Local Level.

### 2.1.4. Orbital Frames

The final relevant frame type is Orbital frames. These are defined by orbital imagery, which must be in a rectangular projection. The details of orbital mapping are described in Section 3.9, but coordinates are defined in the same XYZ system as for Site frames. Orbital frames are indexed in order to support more than one image or map. The ORBITAL(0) frame is special in that it defines a global XYZ coordinate system whose origin is the same as the cartographic latitude/longitude origin.

Important note: PLACES has its genesis in rover operations. Therefore coordinate systems are defined in ways convenient to operations. In the case of orbital frames, this means +X points north ("northing"), +Y points east ("easting"), and +Z points ***down*** (the *opposite* of elevation). This may be extremely surprising to cartographers. Also note, the XYZ coordinates assume a cylindrical projection of the planet; that is, they are not body-centered XYZ but are instead northing, easting, and (negative) elevation values.

### 2.1.5. Reference Frames

All numeric values, whether a position or orientation, must be expressed relative to some other frame. It makes no sense to say the rover is at (10.5, 5.2, 0.1) without also saying how to interpret those numbers. Thus every coordinate or rotation must have a Reference frame, in which the values are expressed.

Generally Rover frames are expressed in the Site frame sharing the same Site index. Site frames are generally expressed in terms of the preceding Site frame. However, these guidelines are not absolute, and any frame can be expressed in terms of any other, as long as a tree structure is maintained (i.e. no circular references). Ground-derived localizations, for example, are often expressed in the global (ORBITAL(0)) frame.

The root of the tree for PLACES is the Root frame, which is defined as the global origin (same as ORBITAL(0). This frame has no numeric location, no solutions, exists implicitly in every view, and is identical across all views. Orbital frames and the landing site (Site 1 for MSL, Site 0 for MER) are all defined in terms of Root. Orbital(0) has a 0 offset and identity quaternion from root.

## *2.2. Rover Motion Counter (RMC)*

The Rover Motion Counter (RMC) is a set of ten indices whose values serve to uniquely identify every location the rover has visited, and thus every instance of the Rover Frame. It also serves to indicate when any mechanism on the vehicle has moved.

The ten indices, in order, are:

Site, Drive, Pose, Arm, CHIMRA, Drill, RSM, HGA, DRT, IC

### 2.2.1. Site Index

The Site index defines which instance of the Site frame is relevant for this RMC. Whenever the Site frame is incremented, all the other RMC values are set to 0. Declaring a new Site frame thus resets all motions and creates a new local area in which to work.

Unlike the other indices, there is no meaning to odd or even values. Also unlike other indices, Sites start with 1 (the landing site), not 0.

### 2.2.2. Drive Index

The Drive index (sometimes called Position) increments whenever the rover drives or otherwise moves its wheels (e.g. trenching or steering). The value is odd while the wheels are actually moving, and even otherwise. Incrementing Drive sets all the other indices (except Site) to 0. Exception: if a mechanism index is odd, indicating it is moving during the drive, then that index is not reset to 0. This should be an unusual case, however. The Drive index is the one of most interest for most localization activities.

### 2.2.3. Pose Index

The Pose index indicates a change in the rover's knowledge of its position or orientation. Unlike the other indices, it does not indicate that anything actually moved; only that the pose knowledge has changed. This may be due to running visodom, doing a sun find, reading the IMU's (Inertial Measurement Units), or receiving an explicit ground command to update the pose knowledge. Note that the rover might actually have moved slightly, mostly in orientation, due to being pushed by the arm (e.g. during a drill preload). There is unfortunately no reliable way to distinguish this from an attitude update, although the arm index will change during arm movements. So a pose change does not indicate arm-induced motion, but lack of an arm update does rule it out.

Pose will be odd while an update is actually being made (including extended IMU updates), even otherwise. Incrementing Pose has no effect on other indices.

#### 2.2.4. Mechanism Indices

The remaining RMC indices (Arm, CHIMRA, Drill, RSM, HGA, DRT, IC) are associated with mechanisms on the rover. They increment to an odd number when the mechanism is moving, and again to an even number once the mechanism stops. More than one mechanism may be moving at the same time. More information about these indices can be found in Section 4.7

The MSL arm has the strength and mass to move the rover (in tilt, primarily) when it is used. Thus the fact that the arm moved, as reflected in the RMC, means that the rover's physical pose may have changed, even if its knowledge has not. This is sometimes relevant and sometimes not, but is the primary reason for maintaining the mechanism RMC indices. The other mechanisms could in theory similarly move the rover, but are unlikely to in practice.

#### 2.2.5. RMC as a Clock

The RMC can be thought of as a clock, albeit one that ticks at irregular intervals. It measures time not as a duration, but in terms of motions.

Given any two valid RMC's, one can determine unambiguously which occurred earlier in time, because the counters only count upwards (except when reset to 0 by incrementing Site or Drive). If the Sites are different, the lowest is earlier. If the same, then the lower Drive is earlier. If those are the same, then consider the remaining indices. If any are lower, then the entire RMC is earlier. Note that it is not valid for some of these indices to be lower and others higher - that indicates the RMC value was corrupted, possibly during transmission.

#### 2.2.6. RMC values of -1

In the PDS archive data, the value -1 sometimes appears for an RMC index. This indicates "unknown" or "does not matter". For example, the Pose RMC index is usually -1 for localization solutions, since the Pose RMC index is meaningless for localizations. Mechanism tables often have -1 in RMC slots other than the mechanism of interest, as the other mechanisms are not relevant. URL's appearing (as href's) in results contain -1's, since the active database interprets -1 as a wildcard for queries.

The value -1 never appears as part of an actual RMC, thus its use as a wildcard flag value by PLACES.

### *2.3. Rover Behavior during Drives and Arm Moves*

The interaction of the RMC with driving and arm motion is relevant to localization, so it is described here.

#### 2.3.1. Driving

Before a drive starts, the Drive index is incremented to an odd value, where it stays throughout the drive. Once the drive ends, the index is incremented to even. During the drive, the rover position and orientation are computed and propagated by the flight software at an 8Hz rate. This means that the same RMC, with an odd-valued Drive index, can actually refer to many different places along the drive.

For this reason, odd-drive RMC's (called dynamic RMC's by PLACES) are insufficient to specify a location; actual time must be used instead. However, very few other activities that require localization (such as imaging) happen during a drive, so this is rarely an issue. To handle the instances where it is (such as taking video while driving), PLACES will store all pose updates it sees in telemetry during a drive, for use with time-based queries. NOTE: currently this is not the case; dynamic (odd-drive) RMC values are not regularly stored. One may be seen occasionally however, as reported in some data product.

Note that the Pose index does not increment during the drive, despite the fact that pose knowledge is changing.

If visual odometry is run to correct the rover's position after a drive, the drive index is first incremented to even. Pictures are then acquired and analyzed onboard the rover. Once a visodom solution is computed, the Pose index is incremented to odd, the value is stored, and Pose is incremented to even again. If the visodom image is downlinked, it will contain the position computed at the end of the drive, not the visodom solution.

### 2.3.2. Arm Operations

Because the arm has the possibility of moving (tilting) the rover, special precautions may be taken during arm operations. If enabled, the Pose counter is first incremented to odd. Arm operations then commence. While Pose is odd, the rover attitude is continually monitored via the IMU and updated at an 8Hz rate. Once the arm motion stops, Pose is incremented to even and the updates stop. Note that the Arm index would have incremented as well due to its motion, along with potentially those of other mechanisms.

This is similar to the behavior during driving, and RMC's with odd Pose values are also called dynamic RMC's. However, arm operations often take longer, and there are more likely to be other activities occurring at the same time. Thus it is more likely to see an odd Pose value in other data, such as imagery. Also, the rover position is not updated during arm ops; only the orientation is. PLACES will again store orientations seen in telemetry during moves, for use with time-based queries (NOTE: as with drives, dynamic orientations are not currently stored on a regular basis).

Not all arm ops will monitor the pose. If not, attitude knowledge will not change (even if the rover moves), meaning the Pose index will not change.

## *2.4. Saved Frames*

The rover has the concept of Saved Frames. This means that the Rover and Site frames at a specific RMC are saved onboard for later use. Saved frames have a single index, which can be positive (an absolute saved frame number) or negative (relative to the most recent saved frame).

A Saved frame is really nothing more than a shorthand to specify the entire 10-element RMC value that was saved. For that reason, Saved frames play no role in PLACES; the full RMC will be used instead. However, as a convenience, PLACES maintains a table mapping saved frame numbers to their corresponding RMC (see Section 4.8.2).

## *2.5. Localization Methods*

Localizations can come from many sources, using many methods. The following list is not comprehensive, but is intended to illustrate what is meant by localization. Note that all onboard methods increment either the Drive or Pose indices.

### 2.5.1. Onboard Methods

- Wheel rotation integration. This is the primary method for the rover to compute its position onboard, and happens during all drives.
- Visual odometry (visodom). This is a more accurate onboard method for position estimation, which analyzes images taken before and after a drive to compute how the rover moved.
- IMU. Onboard intertial measurement units allow continuous estimation of orientation.
- Sun find. A more accurate method of estimating orientation on board, which takes images of the sun to compute how the camera, and thus the rover, is oriented.

### 2.5.2. Ground-Based Methods

- Image analysis. This is the process of analyzing images taken before and after a drive (or series of drives) to refine the pose. Similar to visodom, but likely to be more accurate in ground processing.
- Target updates. An interactive process by which a predefined target is adjusted to match new imagery, thus creating a pose update.
- Orbital localization. Determining where the rover is in an orbital image, either by seeing it directly or (more often) by triangulating from landmarks. Can also be used for orientation, although this is not common.
- Radio tracking. The Navigation team can use UHF two-way Doppler tracking of the rover to estimate rover position in the Mars inertial system, which can be converted to the Mars body-fixed or orbital-image reference frames used by PLACES. This is generally only of use during the descent; other methods are more accurate during surface operations.
- Bundle adjustment. A process that gathers together lots of information and runs a global minimization function to adjust many results simultaneously in order to obtain the best overall solution.

# 3. PLACES Localizations

This section covers the fundamental concepts of PLACES, as they relate to localization and the PDS delivery.

## *3.1. Immutability of Results*

One of the top-level requirements of PLACES is to guarantee immutability of results, except in certain defined cases. That means that an identical query will return the same answer throughout time. This can be important for tactical planning, in order that ongoing localization activity does not disrupt target definitions.

Every query to PLACES involving coordinate transforms returns a flag that indicates whether or not the result is immutable.

Metadata (for example, who created a solution) which do not affect any numeric results is *not* immutable; it can be changed at any time. It is also nullable, so it can be filled in later or left out.

Note that in practice during operations, immutability has not been an issue; all views have remained open and thus mutable. It is mentioned, however, because some of the output contains a flag indicating the immutable status of the data.

## *3.2. Frames*

A "frame" (coordinate frame) is what is defined by PLACES. Frames are identified by a frame type and an instance, which may be an RMC, time value, or index. Note that a frame is not the same as a solution. Frames unambiguously exist; it's their location and orientation (contained in a solution) that is uncertain.

PLACES defines five frame types:

- Site
- Rover
- Orbital
- Local_Level
- Root

All of these except root correspond to the coordinate frames described in Section 2.1. It is the combination of frame type and associated instance identifier (RMC, time, or index) that specifies what is being defined. For example, a Site frame requires an index, thus Site 5 is a frame. So is Rover (5,12,2,6,100,8,0,0,12,2), Rover at SCLK 410455612, Rover (5,12), or Orbital image 3. Local_Level frames are not seen in the PDS extraction.

The Root frame is used as the single reference from which all others are ultimately derived, and corresponds to ORBITAL(0). See Section 2.1.5.

## 3.3. Solutions

Solutions are the currency of PLACES. Localization is not an exact science, and multiple estimates of the rover pose from multiple places can be simultaneously valid for different reasons. Each of these estimates is called a "solution".

A solution is a single estimate of position or orientation for a single frame, relative to some reference frame. Once a solution is defined, it may never be changed for any reason.

Each solution is named by a solution ID. Solution ID's may be shared among multiple solutions, but the pair (Frame, ID) must be unique. One frame can have several solution ID's associated with it, and one solution ID can be re-used for multiple frames.

What a solution really does is to define the relationship between two frames - the one that's part of the solution name, and the one named by the Derived-from frame (not the reference frame; see Section 3.7). If the Derived-from frame moves relative to some third frame, the solution's frame may move as well relative to that third frame - but the relationship between the two is fixed. For example, if someone redefines a Site frame, the Rover frames inside it retain the same relationship to the Site, but may have a different relationship to an earlier Site. In other words, solutions exist in isolation.

## 3.4. Views

Working with multiple, possibly contradictory, solutions for the same frame would be extremely difficult. In addition, in order to make use of the information contained in a solution, it must generally be placed into a broader context, containing relationships to other frames, and thus other solutions. One generally needs a solution ID for each frame in the frame tree of interest, which is not practical to specify directly.

For these reasons, PLACES has a concept of named Views. A View is simply a way of looking at the database that assigns one and only one solution to each frame. Which solution that is, will depend on how the particular View is defined.

Thus, a View is simply a list that specifies which one solution ID to use for each frame. It's a shorthand, relieving the operations user of the burden of specifying a huge list of solution ID's every time they want to use the database.

Views are generally defined in terms of a parent View, with changes applied. For example, if Joe likes Sally's View but disagrees with a couple of rover positions, Joe could create a new View with Sally's as a parent, and then redefine the positions he disagrees with.

Views are thus logically dense (defining a solution for every frame), but are physically sparse (containing just those solutions that differ from the parent).

Views can be created by any operations user, and users can control who is able to modify their view. However, all views are public - there is no restriction on anyone reading a View. The intent of the PDS delivery is to present all non-trivial views that have been created in operations (see Section 4.4).

## 3.5. Positions vs. Orientations

Positions (offsets in XYZ space) and orientations (attitude of the rover) are treated completely separately by PLACES. Within one View, a given frame can have one solution ID for the position and a different one for the orientation. Although both are needed for a full coordinate frame transform, they are very commonly derived separately, so a full separation makes it easier to work with them. This document often talks about positions for convenience, especially in terms of coordinate transforms. In most cases, the concept applies equally well to orientations (where an offset maps to a rotation and adding offsets maps to compositing rotations).

## 3.6. Metadata

Most resources have associated with them a set of metadata. This metadata include things like who created the resource, why it was created, a description of it, who can update it, etc. Use of these metadata is not mandatory but it is highly recommended for documentation. For example, a solution ID might be created with user=”rgd”, institution=”JPL-MIPL”, method=”marsnav image analysis”, and description=”Compensate for wheel slippage during rover bump in the middle of the BigCrater panorama”. Such metadata help users determine which solutions or views they want to use.

These metadata are dumped to XML files as part of the PDS delivery (see Sections 4.4.2.1 and 4.6.2).

## 3.7. Understanding the Derived-from Frame

This section talks about positions/offsets for convenience; it applies equally well to orientations/rotations.

Localization solutions have to be derived from something. They specify a new offset or rotation relative to some other frame. This frame need not be the same as the Reference frame! In other words, the source of the localization (the Derived-from frame) does not have to match the way in which the numbers are expressed (the Reference frame). See Figure 1.

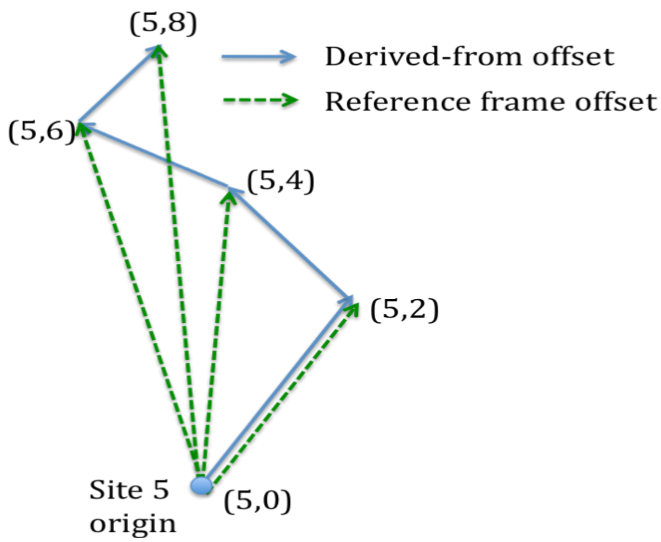


**Figure 1: Derived-from and Reference frames for a drive sequence**

The archetypal example of this is the case of the rover counting wheel rotations onboard. As the rover drives, it counts wheel rotations and integrates them in order to determine how far it went. All the rover can really determine is how far the *last* drive arc was. But, the position is *expressed* in the Site frame. Consider the example in Figure 2. Let's say that for site 5, drive 12 is located at 5.0 meters (consider only the X coordinate for clarity). The distance from drive 12 to 14 was 2.0 meters, and from drive 14 to 16 was 1.0 meters. That means frame (5,14) has a derived-from frame of (5,12), and a location of 7.0 meters relative to Site 5. Frame (5,16) is expressed as 8.0 meters from Site 5, even though it's derived-from frame is (5,14). Now say it is determined later that the drive from 12 to 14 was shorter than expected due to wheel slippage, for a distance of only 1.5 meters instead of 2.0. This is shown on the right side of Figure 2. The location as measured in the Site 5 frame for (5,14) is now 6.5. But the drive from 14 to 16 is still valid, at 1.0 meters. That means the position of (5,16) as measured in Site 4 should now be 7.5 meters instead of 8.0. The relationship between frame (5,16) and its derived-from frame (5,14) did not change - it's still 1.0 meters. But the relationship between (5,16) and its Reference frame (Site 5) did change.

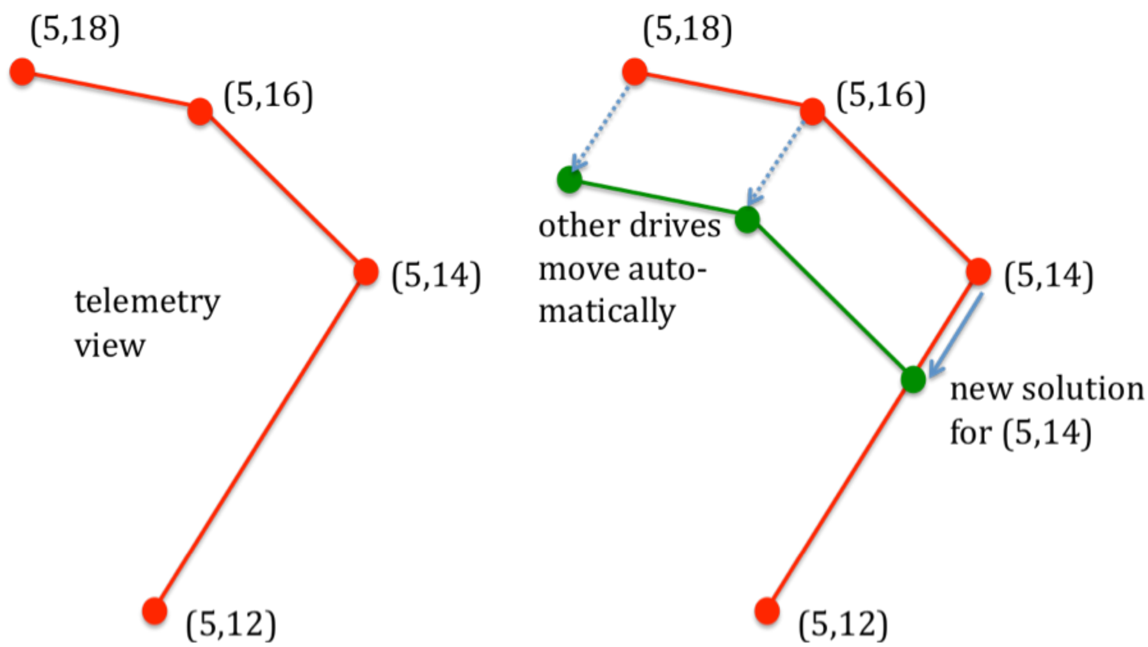


**Figure 2: Result of moving one solution in a chain of Derived-from frames**

This is the key point: for a given solution, the numeric relationship between the frame and its Derived-from frame does not change - even if the numbers are expressed in a different Reference frame. If the Derived-from frame moves in a different view, the solution frame moves along with it.

The active PLACES database keeps track of this and applies the appropriate corrections when queries are requested. In the PDS delivery, the Global Frame (Section 4.4.2.3) and CSV (Section 4.4.2.4) data have had these corrections applied. However, it is an important issue to understand if you want to make use of the Raw data (Section 4.4.2.2) directly.

### 3.7.1. Derived-from and Sites

When a Site frame is declared, the position of that site frame with respect to the previous frame is saved and downlinked as a site offset. These site offsets provide a link between each site and its predecessor.

But how is that site offset derived? It's derived as the position of the rover at the time the Site frame is declared. So in reality, the Derived-from of a Site frame should be the last Rover frame before the Site was declared.

However, that is impractical for two reasons. First, the ultimate Rover frame at the time the site is declared may not (ever) be known, depending on how much data were telemetered. In the case of rover drives, we know the Derived-from is drive n-2; if n-2 is not present in the database, we can use the most recent prior drive with no loss of accuracy. However, in the case of a new site, we may not know the proper drive index to use - and picking the wrong one would lead to incorrect results.

A second factor is efficiency - it is not practical to check every single drive throughout the entire mission when doing coordinate transforms. Once the derived-from chain gets to a site frame, site-to-site transforms can be used instead.

For these reasons, the Derived-from for a Site frame for "telemetry" is set to the previous Site frame (just as its Reference frame is).

The implication of this is that localizations within a site will not affect anything in the next site without explicit user action. This actually matches typical user expectations better, since sites are often considered to be independent realms.

### 3.7.2. Derived-from and Views

There is one other important factor to note regarding Derived-from and Reference frames: they must use the same View. This view is attached to the Reference frame by convention, but it applies equally to both. As described above, solutions are ultimately described by the offset from the Derived-from to the solution frame. This offset is computed, if possible, when the solution is submitted. Computing this offset requires determining the position of the Derived-from frame in the Reference frame using the given view. Once this offset is computed, the reference view is no longer needed.

### 3.7.3. Derived-from and Orientations

The above discussion of the Derived-from frame centered around position. Orientations also have a Derived-from frame, which should be used in the same way if the orientation is derived from image analysis.

However, most orientations are determined using an IMU or a sun-find. These methods of determining orientation provide what might be thought of as absolute orientations but are actually relative to the surface (north/down). For this reason, these solutions generally use the Site frame as their derived-from. Orientations derived from looking at orbital imagery would use an orbital frame as their derived-from.

### 3.7.4. Parent Views

Every view has a parent view, which is used to fill in the sparse RMC list of the child and make available a solution for every RMC. This is true until the root of the view tree is reached.

The root of the View tree is "telemetry". The "telemetry" view has no parent; it is expected to provide a value for every known RMC. It may in fact have a few holes (such as unsolicited moves) which are filled in by the child views, but sibling views not containing the unsolicited move will simply not see the missing RMC.

## *3.8. Are Sites Parallel?*

The question as to whether site frames are parallel to each other is somewhat controversial within the MSL project. On the one hand, sites are defined as +X pointing north and +Z pointing down according to the local gravity vector. On a curved surface such as Mars, this means site frames are not parallel; they mimic lines of longitude, which are not parallel to each other.

On the other hand, within the range of locations relevant to operations, the difference is negligible. The difference in Z due to planet curvature is on the order of 15cm over a 1km traverse, far less than the measurement error over that distance. Therefore, the rover onboard assumes site frames are parallel.

This is reasonable for operations; it is rare that data from over (or even close to) 1km away are used during operations. However, for precision large-scale mapping, it could make a difference.

PLACES remains agnostic on this issue, letting the user decide. The Site frame stores, like any other frame, both a position and an orientation relative to its reference frame. For the "telemetry" view, the orientation quaternion is set to identity (no rotation relative to the reference), which mimics the onboard software and makes all sites parallel. Users are free, however, to define a site solution which contains a rotation term, in order to compensate for planet curvature. If such solutions are defined, they will show up in the PDS dump.

It should be noted that in practice, large-scale mapping to date has been done almost entirely using the Global frame. This renders the question of Site frame parallelism moot, since Site frames are not used. Planet curvature is accounted for by the map projection (Section 3.9.1).

## *3.9. Mapping of Orbital Images*

Orbital images are referenced using a single index. Each orbital image stands on its own, defining a coordinate system relative to some parent (often the global frame, orbital(0), but possibly other orbital images).

Data associated with an orbital image include:

- Image identification (filename or URL)
- Pixel scale
- Size in pixels
- Location on the image (in pixel space) of the XYZ origin
- Standard descriptive metadata information (originator, description, etc.) as with any other solution ID or view.

These data, except for the descriptive metadata, are immutable once set.

Additionally, like any other frame, orbital images contain:

- Reference frame identification
- Transform from XYZ origin to the reference frame origin
- Derived-from frame (should be the same as reference frame)

The transform can include an offset and/or orientation just like any other frame reference. This allows multiple maps to all have North up yet have a rotation between them to cover large distances (see Section 3.8). This rotation should be identity (no rotation) in most cases, although rotation between Orbital frames is supported.

### 3.9.1. Orbital Image Map Projection

Orbital images (or more precisely, orbital image coordinate frames) must be in a rectangular projection, with north up in the image. The line dimension of the image is aligned north/south with the sample dimension aligned east/west. There is a constant pixel scale (meters/pixel) in both directions, which is defined in the orbital image metadata.

While simple rectangular projections are not generally considered good for mapping, the landing site and nominal science area in Gale crater is < 5 degrees from the equator and therefore shapes, areas, and distances are reasonably true. Angles between two points are true. As noted below, the east-west distortion at this latitude is trivial compared to the expected error in local terrain measurements near the rover and in distance measurements during its nominal traverse to Aeolis Mons (aka Mt. Sharp).

### 3.9.2. Projected Meters vs. True Meters

It is important to note that the east/west measurement, called "easting", is defined by meters *at the equator*. Orbital images as used by the MSL localization science team, and thus by PLACES, use a rectangular projection, so that lines of longitude are parallel (as are lines of latitude, naturally). For convenience there are an equal number of meters between each line of longitude, regardless of latitude. This greatly simplifies calculations, and as a nice side-effect causes site frames to be parallel in the orbital projection.

These "easting" meters at a given latitude are related to true meters at the equator by the simple formula:

$$map\_meters = \frac{true\_meters}{\cos(\phi)}$$

where $\phi$ is the planetocentric latitude. Since MSL landed at about -4.6 degrees latitude, the difference is on the order of 3mm per meter.

Meters measured in the north-south direction ("northing") are true meters.

It should be noted that the rover works only with true meters, and does not take into account planet curvature. Thus the "telemetry" view (and only the "telemetry" view) is actually expressed in true meters, and assumes a flat world. The localization error in telemetry, even with visual odometry turned on, is far greater than the true vs. easting meters difference, so in practice this distinction makes no difference.

### 3.9.3. Elevation, Z, and Radial Distance

As noted in Section 2.1.4, Z values are measured in a +Z-down coordinate system. They are thus the negative of cartographic elevations, which are measured +Z up. Throughout this document and the data, "Z" and "elevation" are used carefully to indicate the sign of the associated number. Note that for MSL's entire area of operations, the Z value is positive, while elevations are negative (thus, the ground is below the reference surface). But raises the question, what is the reference surface for these elevations? It is an important question if one needs to convert to radius values (distance from the center of the planet).

Elevations in PLACES follow the conventions established by the MSL Localization Science team. They are referenced to the MOLA areoid (geoid in Earth terms). The areoid defines "sea level" on Mars, which is the level water would actually reach if Mars were filled with water. The areoid takes into account variations in local gravity, so it varies from the ellipsoid depending on where the rover is. The difference between the areoid and its 3396 km reference ellipsoid varies from -365 to -368 m across the current traverse; it could approach -379 m if the rover climbs Aeolis Mons (Mt. Sharp). In addition, the areoid is based on a 3396.0 km reference ellipsoid, while the localization team uses 3396.190 km. This 190 m difference needs to be accounted for.

A full treatment of the MOLA areoid is beyond the scope of this document; it is defined by [ 5 ].

As a convenience to PDS users, several areoid values (differences vs. the ellipsoid) along the traverse (sols 331-705) have been extracted, and are presented in Table 1. A close approximation to the areoid can be derived as a linear function of latitude only. Although the data was extracted from sols 331-705 only, the areoid varies slowly enough to make the fit valid for most purposes over the entire traverse. Figure 3 shows this function and the curve fit. The correction is:

$$\Delta A = 44.603 * \phi_{pc} - 350.15$$

where $\Delta A$ is the delta areoid value and $\phi_{pc}$ is the planetocentric latitude, computed by:

$$\phi_{pc} = \frac{x}{R_e} \bullet \frac{180}{\pi}$$

where $x$ is the global-frame coordinate from PLACES, and $R_e$ is the ellipsoid radius, or 3396190 meters.

Distance from the planet center ( $R$ ) can then be computed given the PLACES global-frame z value:

$$R = -z + \Delta A + R_e$$

| Longitude | Latitude | MOLA Areoid (m) 3396 km reference | Corrected Areoid (m) 3396.19 reference |
|---:|---:|---:|---:|
| 137.445394 | -4.59051 | -364.795 | -554.795 |
| 137.444891 | -4.59298 | -364.925 | -554.925 |
| 137.424 | -4.609486 | -365.815 | -555.815 |
| 137.41785 | -4.619789 | -366.333 | -556.333 |
| 137.410291 | -4.625275 | -366.562 | -556.562 |
| 137.400206 | -4.632524 | -366.831 | -556.831 |
| 137.399178 | -4.64012 | -367.179 | -557.179 |
| 137.384657 | -4.647657 | -367.309 | -557.309 |
| 137.382585 | -4.659938 | -367.913 | -557.913 |

**Table 1: MOLA Areoid Selected Values**

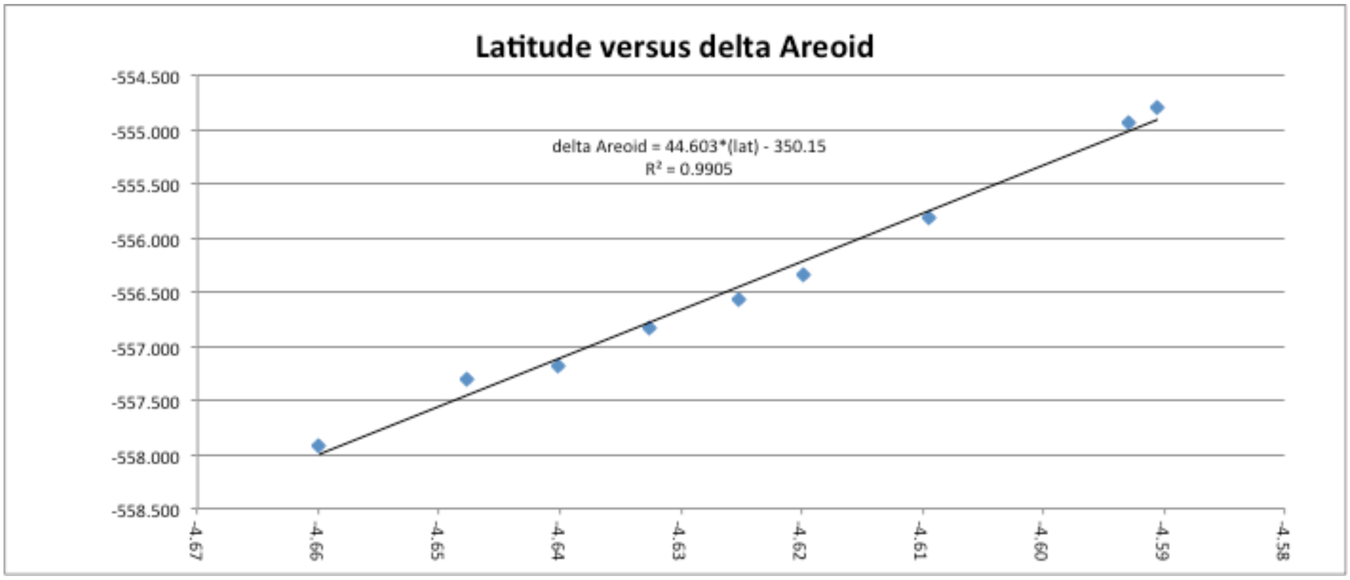


**Figure 3: Latitude vs. Delta Areoid**

### *3.10. Sources of Telemetry Data*

The "telemetry" view will be filled in using as many data sources as possible. Any telemetry channel that provides position or orientation value for a given RMC is a potential source. The specific list of sources may expand over time, but currently include:

- Image data (using the labels, derived from the Image Data Product Header (IDPH))
- Non-image data products (using the common telemetry product header)
- Kinematic state data products: general, and specific to arm, HGA, RSM, and mobility.
- EVR's

# 4. PDS Delivery Contents and Structure

The PDS delivery of PLACES data consists of 5 sections. These sections are:

- Localizations
- Maps
- Orbital Metadata
- Mechanisms
- Ancillary Tables

## *4.1. Sols Covered by a Delivery*

PDS deliveries are typically defined by a sol range, for example 0-90, 91-120, 121-180, etc. However, PLACES works primarily in the RMC realm of Sites, rather than Sols. For that reason, any given PDS PLACES distribution will contain all data for a range of Sites. This range of Sites is large enough to encompass the entire range of Sols. But it does mean that some additional data from outside the nominal sol range is necessarily provided as part of the delivery.

## *4.2. PDS Volume Structure*

PLACES data is delivered in the following structure:

```
<root>
| -- AAREADME.TXT
| -- <CATALOG>
        | -- CATINFO.TXT
        | -- DSMAP.CAT
        | -- MSL_INSTHOST.CAT
        | -- MSL_MISSION.CAT
        | -- MSL_PLACES_DS.CAT
        | -- MSL_REF.CAT
        | -- PLACES_PERSON.CAT
        | -- PLACES_REF.CAT
| -- <DATA>
        | -- <MAPS>
                | -- msl_orbital_dem.lbl
                | -- msl_orbital_dem.img
                | -- msl_orbital_map.img
                | -- msl_orbital_map.lbl
        | -- <LOCALIZATIONS>
                | -- localized_interp.csv
                | -- localized_interp.lbl
                | -- localized_interp_demv2.csv
                | -- localized_interp_demv2.lbl
                | -- localized_pos.csv
                | -- localized_pos.lbl
                | -- localized_pos_demv2.csv
                | -- localized_pos_demv2.lbl
                | -- telemetry.csv
```

```
                | -- telemetry.lbl
| -- <DOCUMENT>
        | -- DOCINFO.TXT
        | -- MSL_CAMERA_SIS.LBL
        | -- MSL_CAMERA_SIS.PDF
        | -- MSL_EDR_VOLSIS.PDF
        | -- MSL_EDR_VOLSIS.LBL
        | -- ODL.TXT
        | -- PLACES_PDS_SIS.PDF
        | -- PLACES_PDS_SIS.LBL
        | -- PPPCS_VOL9_MAY_13_2016.LBL
        | -- PPPCS_VOL9_MAY_13_2016.PDF
        | -- VICAR2.TXT
| -- ERRATA.TXT
| -- <EXTRAS>
        | -- EXTRINFO.TXT
        | -- <XML_ANCILLARY_DATA>
                | -- <ancillary>
                        | -- *.xml
                | -- <localizations>
                        | -- *.xml
                        | -- <localized_interp_global_frame>
                                | -- *.xml
                        | -- <localized_interp_raw>
                                | -- *.xml
                        | -- <localized_interp_demv2_global_frame>
                                | -- *.xml
                        | -- <localized_interp_demv2_raw>
                                | -- *.xml
                        | -- <localized_pos_global_frame>
                                | -- *.xml
                        | -- <localized_pos_raw>
                                | -- *.xml
                        | -- <localized_pos_demv2_global_frame>
                                | -- *.xml
                        | -- <localized_pos_demv2_raw>
                                | -- *.xml
                        | -- <telemetry_global_frame>
                                | -- *.xml
                        | -- <telemetry_raw>
                                | -- *.xml
                | -- <mechanisms>
                        | -- *.xml
                | -- <orbital>
                        | -- *.xml
| -- <INDEX>
        | -- INDXINFO.TXT
        | -- INDEX.TAB
```

```
        | -- INDEX.LBL
| -- VOLDESC.CAT
```

The files named in UPPER_CASE are standard PDS 3 files, or documentation files, and are not explained further in this document.

The directories are named in <BRACKETS>; the <> do not actually appear in the volume. So for example the orbital map is in the relative path “DATA/MAPS/orbital_map.img”.

Due to PDS 3 requirements, the XML files are in the EXTRA directory, rather than DATA. It is important to note that the XML files are the *primary* source of information, as they are what was extracted directly from the database. The files in DATA, while generally more useful than the raw XML files, are derived from the XML (with the exception of the map).

Within the DATA and EXTRAS directories are 6 major directories; each is described in the following sections (both the DATA and EXTRAS localization directories are described in the same section).

```
root/DATA/LOCALIZATIONS
root/EXTRAS/XML_ANCILLARY_DATA/localizations
root/DATA/MAPS
root/EXTRAS/XML_ANCILLARY_DATA/orbital
root/EXTRAS/XML_ANCILLARY_DATA/mechanisms
root/EXTRAS/XML_ANCILLARY_DATA/ancillary
```

Many of the files are in XML format. In order to present the data as originally entered, and reduce the possibility of translation error, the XML files are extractions directly out of the operational PLACES database. A few elements that are not relevant to the archive are removed, such as href tags, the <?xml> boilerplate, xmlns (namespace) attributes, and redundant sclk values (those in scientific notation). But the files are not otherwise edited.

Note that per PDS3 requirements, all text files (everything except the .img) in the DATA directory, as well as all standard PDS3 files, use CR/LF as a line terminator. The files in EXTRAS use the Unix standard of LF only.

## *4.3. Data Formats and Ranges*

Unless otherwise specified in the detailed description, the following data ranges apply to all data items of the given type.

`X, Y, Z values`: Double-precision floating point. Data range is unlimited. Measured in meters.

`Northing, easting, elevation values`: same as XYZ.

`Quaternions`: Quaternions represent a rotation in 3-D space, in the order (s, v1, v2, v3), where:

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

$$v_n = \sin\left(\frac{\theta}{2}\right) * a_n$$

and $\theta$ is the angle of rotation and $a$ is a unit vector defining the axis of rotation. All quaternions are unit quaternions, thus the data range is -1.0 ... +1.0.

`RMC components`: Integers starting at 0 with no specific upper bound, with -1 indicating no value.

`Angles`: Represented as floating-point numbers in degrees, typically -360.0 to +360.0. Some angles have different valid ranges, including > 360.

`Latitude`: Double-precision floating-point, measured in degrees from -90.0 to +90.0.

`Longitude`: Double-precision floating-point, measured in degrees from -360.0 to +360.0.

`Pixel values`: Floating-point values with no specific range (pixel coordinates may be off a given image). Note that the center of the upper-left pixel of an image has coordinates line=1.0 and sample=1.0 for 1-based coordinates (the default), and line=0.0, sample=0.0 for 0-based coordinates (when specified).

## *4.4. Localizations*

A large set of data is provided for each View that is extracted for PDS. The format and layout of these data is standardized; i.e. it is the same for each View.

### 4.4.1. Included Views

The set of views that are included may change from delivery to delivery. However, the current set is described below, and consists of 5 views.

#### 4.4.1.1. telemetry

The "telemetry" view contains locations and orientations as reported by the rover. They are not corrected on the ground and thus represent the raw data from which other localizations are derived. The effects of wheel slippage and other cumulative errors mean that, in a broad sense, the telemetry positions do not match any map or other reference. The telemetry view should thus not generally be used directly; one of the other views is almost always better. However, this view is of considerable historical interest because it preserves the state of rover knowledge, before ground adjustment. Also, in a local sense, such as within one sol's worth of driving, the telemetry view does a reasonable job of showing the relative traverse for the day. Finally, comparing telemetry to a localized view can provide some idea of how much the rover slipped.

The rover is capable of updating its pose knowledge, for example by using visual odometry, reading the IMU, etc. These updates are tracked using the Pose counter in the RMC (see section 2.2.3). The pose counter acts like a solution ID in that different pose values represent different states of pose knowledge by the rover. A complete treatment of how Pose should be handled is beyond the scope of this document. Nevertheless, a useful rule of thumb is to use the highest numbered Pose for any given Site/Drive. This reflects the results of visodom and other updates. Unfortunately the reason for any given Pose update is not available to, nor tracked by, PLACES.

#### 4.4.1.2. localized_pos

The "localized_pos" view contains locations as updated by the MSL Localization team. This team, headed by the MSL Localization Scientist, determines updated positions for the rover after every drive by comparing orthorectified images from the navcam with satellite (HiRISE) photos. These locations are specifically designed to match the map and DEM cutouts (Section 4.5) included with this delivery. The localized positions have an expected accuracy of 0.5m compared to the base map.

The localized_pos view is sparse, containing just those locations where localization was performed (typically the last drive of a Sol, although localization is occasionally done more than once on a Sol). It does not contain intermediate positions during a drive.

Z values (negative of elevation) for each location are derived from the DEM for the base map, created from orbital HiRISE data (see Section 4.5). The Z value is simply the negative of the DEM value at the given XY coordinate (because the +Z axis is down). The telemetered Z value (relative to the Site origin) is ignored for this purpose.

The localization team does not currently adjust orientation. Thus, the orientations included in the solution are copied straight from the telemetry view.

#### 4.4.1.3. localized_interp

The "localized_interp" view contains all the localized solutions that are in localized_pos. In addition, it includes "interpolated" values for all drive segments in between those locations. So every site/drive location seen in telemetry, should have an entry in localized_interp, creating a continuous drive path.

Interpolation proceeds as follows. Given two endpoints (localized positions from localized_pos), it gathers all telemetry entries (highest pose only) between those points. The start is subtracted from the end in both localized_pos and telemetry views to get vectors. The vectors are then compared to determine an angle of rotation between them and a scale factor. These values are then used to transform each telemetry point:

$$\vec{V}_t = \vec{F}_t - \vec{O}_t, \quad \hat{V}_t = \frac{\vec{V}_t}{\left\|\vec{V}_t\right\|}$$

$$\vec{V}_{lp} = \vec{F}_{lp} - \vec{O}_{lp}, \quad \hat{V}_{lp} = \frac{\vec{V}_{lp}}{\left\|\vec{V}_{lp}\right\|}$$

$$\theta = \cos^{-1}\left(\hat{V}_t \cdot \hat{V}_{lp}\right)$$

$$\hat{A} = \hat{V}_t \times \hat{V}_{lp}$$

$$Q = \left(\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)\hat{A}\right)$$

$$s = \max \frac{\left(\vec{V}_{lp}\right)_{x,y,z}}{\left(Q * \vec{V}_t\right)_{x,y,z}}$$

$$\vec{P}_{li} = s\left(Q * \left(\vec{P}_t - \vec{O}_t\right)\right) + \vec{O}_{lp}$$

Where

$\vec{O}_t, \vec{F}_t$: original and final points, in telemetry
$\vec{O}_{lp}, \vec{F}_{lp}$ : original and final points, in localized_pos
$\theta$ : angle by which path is rotated from telemetry to localized_pos
$\hat{A}$ : axis of rotation
$Q$ : quaternion expressing rotation from $\hat{V}_t$ to $\hat{V}_{lp}$
$s$ : scale factor for path
$\vec{P}_t$: Point to convert, in telemetry
$\vec{P}_{li}$ : Final converted point, for localized_interp

This method preserves a continuous path based on the shape of the path in telemetry. It implicitly assumes all errors are evenly distributed along the path. This is of course not true; therefore the interpolation is an approximation. However, the true distribution of errors is not known, so no exact answer is possible.

Note that Z values (negative of elevations) are interpolated along with the X and Y positions. This means the localized_interp Z value will follow the curve defined by telemetry for Z, while meeting the DEM at the endpoints. These are higher resolution data than is available from the orbital DEM and should therefore track the surface better. However, the endpoints are constrained to DEM elevations and this may introduce some distortions in the Z shape at the endpoints.

As with localized_pos, orientations are not currently adjusted as part of this process, and are copied directly from telemetry.

#### 4.4.1.4. localized_pos_demv2

Prior to PDS Release 11, the MSL Localization Team used the so-called "version 2" DEM. This version, while usable, had a number of discontinuities in the DEM at seams between source images. These seams were corrected in a new release, called "version 3". There is no difference in the map itself; only the elevation values in the DEM changed. Version 3 was fully implemented in operations on February 3, 2016, at which time the entire database was updated to consistently use v3 DEM values. Thus, the primary views as of PDS release 11 use exclusively v3 elevations. However, the elevations using the v2 DEM were saved in *_demv2 views. These elevations are reported in separate views in PDS mainly for historical interest, but also to facilitate comparison with data in older PDS releases.

The localized_pos_demv2 view thus contains exactly the same data as the localized_pos view, except that the elevations come from the v2 DEM.

#### 4.4.1.5. localized_interp_demv2

Analogous to the previous section, the localized_interp_demv2 view contains exactly the same data as the localized_interp view, except that the elevations are from the v2 DEM.

#### 4.4.1.6. Future Views

In the future, other views are expected. Other localization and interpolation methods are in development. When they are deployed, these views will appear in the PDS delivery as well. In addition, a "best_tactical" view will contain the best (as defined by the ops team) interpolated localizations for every point, regardless of source. This best_tactical view exists now. However, since there is only one active localization and interpolation method, it is currently identical to localized_interp, so it is not being included in the PDS delivery.

### 4.4.2. Data Included in Each View

For each view, four types of information are extracted. Each type is described below.

#### 4.4.2.1. View Metadata

Metadata for the view, as entered by the user who created it, are contained in a single file, "***view***`_metadata.xml`", in the `EXTRAS/XML_ANCILLARY_DATA/localizations` directory. An example is shown below.

```
<view group="telemetry" stable="false" id="telemetry">
    <origination institution="JPL" creation_date="2012-08-02" add_date="2012-08-
02" method="init script" user="hbmorten"/>
    <description>Telemetry View</description>
</view>
```

view: Contains the view name (id), the group it is in (who can modify in the database), and a "stable" flag, which indicates whether or not the view is "closed" to future modification and should always be false for the views submitted to PDS.

`origination`: Contains information about who created the view and when.

description: Contains a short description of the view. Section 4.4.1 has a much more complete description.

#### 4.4.2.2. Raw Data

This directory, named "***view***_raw" under the EXTRAS/XML_ANCILLARY_DATA/localizations directory, contains the data entered into PLACES for the view, in the manner in which it was entered. This means there is no consistency in terms of reference frames. If one localization was done relative to a global frame while the next was done relative to a site frame, those raw values are reported.

The usefulness of the raw data is that it reflects what the user actually entered, before any processing. Thus any bugs that might occur during processing (e.g. to translate to the global frame, below) do not impact this version. Also all the known relationships are preserved, so if the PDS user wanted to e.g. adjust the location of a site, the effects of that on entered solutions could be re-computed. However, the potential inconsistencies in representation make this directory somewhat less useful for the casual user.

Note that for the telemetry view, there are no inconsistencies in representation, since the rover always reports data in the same way. All reference frames are the enclosing Site.

Within the directory, there are three types of files:

```
CreationTime.txt
MSL_view_SiteFrame.xml
MSL_view_RoverFrame_Site_*.xml
```

CreationTime.txt is a text file that simply says when the data presented here was extracted from the database.

There is one SiteFrame file, containing all the Site frame definitions. Then for each Site, there is a RoverFrame file for that site, containing all the positions in the site. Both of these share the same format.

The site and rover frame files consist of a <list> of <solution> entries. An example is presented here:

```
<list>
...
<solution add_date="" immutable="false" solution_id="telemetry">
    <rmc drive="24" frame="ROVER" pose="2" site="18"/>
    <offset x="-3.8948987" y="-0.3826753" z="-0.2227737">
        <solution_metadata>
            <derived_from drive="22" frame="ROVER" pose="-1" site="18"/>
            <reference_rmc frame="SITE" site="18"/>
            <reference_view>telemetry</reference_view>
        </solution_metadata>
    </offset>
    <orientation s="0.18027501" v1="-0.036325052" v2="0.022130704"
v3="0.9826961">
```

```
        <solution_metadata>
            <derived_from frame="SITE" site="18"/>
            <reference_rmc frame="SITE" site="18"/>
            <reference_view>telemetry</reference_view>
        </solution_metadata>
    </orientation>
    <origination add_date="2012-08-02" creation_date="2012-08-02"
institution="JPL" method="init script" user="hbmorten"/>
<solution>
...
</list>
```

`solution:` specifies the add_date, solution_id and immutability flags, none of which are particularly important for the archive. Note that the solution_id is not the same as the view name.

`rmc:` defines the RMC and frame for this solution.

`offset:` provides the XYZ coordinates (position) of this solution (relative to the reference frame).

`orientation:` provides the quaternion (s is the scalar component; v1, v2, and v3 are the vector components) specifying the orientation for this solution (relative to the reference frame). Mathematically the quaternion should be interpreted as follows: Given a vector expressed in the current frame, multiplication by this quaternion will give the same vector as expressed in the reference frame.

`solution_metadata:` specifies the Derived-from frame (Section 3.7), Reference frame (Section 2.1.5), and the view for the reference frame. There are two `<solution_metadata>` entries; one is used for the offset (position) and the other is for the orientation. They need not be the same.

`origination:` specifies metadata for the solution, including when it was created, when it was added to the database, and who created it.

#### 4.4.2.3. Global Frame Data

This directory, named "***view***`_global_frame`" under the `EXTRAS/XML_ANCILLARY_DATA/localizations` directory, contains the data in the raw directory, translated so that the global frame (ORBITAL(0)) is the reference frame. It thus represents a consistently-referenced data set.

Within the directory, there are the same three types of files as with the raw directory, but the contents are slightly different:

```
CreationTime.txt
MSL_view_SiteFrame.xml
MSL_view_RoverFrame_Site_*.xml
```

`CreationTime.txt` is a text file that simply says when the data presented here were extracted from the database.

There is one SiteFrame file, containing all the Site frame definitions. Then for each Site, there is a RoverFrame file for that site, containing all the positions in the site. Both of these share the same format. However, the format of the data is *different* from the raw files (because of the PLACES database query used to extract the data).

The site and rover frame files consist of a `<list>` of `<translate>` entries. An example is presented here:

```
<list>
...
<translate immutable="false">
    <view>localized_pos</view>
    <from-rmc drive="2470" frame="ROVER" pose="-1" site="48"/>
    <reference-rmc date_added="2012-08-02 02:00:45.0" frame="ORBITAL" site="0"/>
    <offset x="-276693.318" y="8142579.427" z="4446.85"/>
    <orientation s="0.73431003" u="-0.04758037" v="-0.030957447" w="0.6764366"/>
    <conversions>
        <latitude value="-4.667983634858819"/>
        <longitude value="137.37024003873523"/>
        <elevation value="-4446.85"/>
        <planetodeticLatitude value="-4.723174850171967"/>
        <ellipsoidRadius value="3396190.0"/>
        <roll value="-0.11201306365777013"/>
        <pitch value="0.01890661113634033"/>
        <yaw value="1.4877357462633636"/>
    </conversions>
</translate>
...
</list>
```

`translate`: specifies the immutability flag. Not particularly important for the PDS archive.

`view`: names the view (which should match the directory name and file name).

`from-rmc`: specifies the RMC for this location. Although not shown in this example, the `date_added` attribute may appear in some views.

`reference-rmc`: specifies the reference RMC, which is always going to be ORBITAL(0).

`offset`: specifies the position (offset) of this frame. Note that x is northing, y is easting, and z is the negative of elevation. So in the above example, the location is 273455.365 meters south of the equator, 8145617.113 meters east of the prime meridian (as measured at the equator, see Section 3.9.1), and 4499.17 meters below the datum reference (i.e. elevation is -4499.17).

`orientation`: provides the quaternion (s is the scalar component; v1, v2, and v3 are the vector components) specifying the orientation for this solution (relative to the reference frame). Mathematically the quaternion should be interpreted as follows: Given a vector expressed in the current frame, multiplication by this quaternion will give the same vector as expressed in the reference frame.

`conversions:` contain a number of conversions of the data before to different useful forms. All except `ellipsoidRadius` are echoed in the CSV file; see the next section for details on these items. `ellipsoidRadius` is the radius of the ellipsoid used in the latitude and longitude calculations.

#### 4.4.2.4. CSV File

The CSV, or Comma-Separated-Value, file will likely be the most useful file for most users. This file is named "***view***`.csv`" in the `DATA/LOCALIZATONS` directory. There is nothing in this file that isn't in (or easily computable from) other files in the delivery, but it is gathered into a more convenient and easy to use format.

The file is a standard text file. There are 22 fields per line, separated by commas. The first line contains column headings; all other lines contain data. The columns are described below. But first, an example, from the localized_pos view:

```
frame,site,drive,pose,landing_x,landing_y,landing_z,northing,easting,
planetocentric_latitude,planetodetic_latitude,longitude,elevation,
map_pixel_line,map_pixel_sample,dem_pixel_line,dem_pixel_sample,roll,pitch,yaw,sclk,sol

SITE,1,-1,-1,0.000,0.000,0.000,-272039.268,8146811.223,-4.589466996,-4.643738049,
137.441632997,-4501.040,2101.07,19932.89,526.19,4983.73,0.000,-0.000,0.000,397502188,0

ROVER,1,0,0,0.000,0.000,0.000,-272039.268,8146811.223,-4.589466996,-4.643738049,
137.441632997,-4501.040,2101.07,19932.89,526.19,4983.73,-2.414,-3.612,112.722,397502188,0

SITE,2,-1,-1,0.000,0.000,0.000,-272039.268,8146811.223,-4.589466996,-4.643738049,
137.441632997,-4501.040,2101.07,19932.89,526.19,4983.73,0.000,-0.000,0.000,397673022,2

ROVER,2,0,0,0.000,0.000,0.000,-272039.268,8146811.223,-4.589466996,-4.643738049,
137.441632997,-4501.040,2101.07,19932.89,526.19,4983.73,-2.422,-3.618,111.519,397673022,2

SITE,3,-1,-1,0.000,0.000,0.000,-272039.268,8146811.223,-4.589466996,-4.643738049,
137.441632997,-4501.040,2101.07,19932.89,526.19,4983.73,0.000,-0.000,0.000,397937710,5

ROVER,3,0,0,0.000,0.000,0.000,-272039.268,8146811.223,-4.589466996,-4.643738049,
137.441632997,-4501.040,2101.07,19932.89,526.19,4983.73,-2.422,-3.618,111.519,397937710,5

ROVER,3,78,-1,0.118,5.987,0.170,-272039.150,8146817.210,-4.589465001,-4.643736030,
137.441734000,-4501.210,2100.60,19956.84,526.07,4989.72,-1.329,5.414,-132.025,398917746,16

ROVER,3,100,-1,3.922,3.140,0.440,-272035.346,8146814.363,-4.589400830,-4.643671107,
137.441685971,-4501.480,2085.38,19945.45,522.27,4986.87,-2.905,4.657,-164.689,399361043,21

ROVER,3,260,-1,3.786,15.384,1.340,-272035.482,8146826.607,-4.589403124,-4.643673428,
137.441892534,-4502.380,2085.93,19994.43,522.41,4999.11,-1.283,-4.294,64.617,399450377,22

ROVER,3,372,-1,1.188,32.502,1.560,-272038.080,8146843.725,-4.589446954,-4.643717772,
137.442181325,-4502.600,2096.32,20062.90,525.00,5016.23,-1.370,-1.650,116.500,399622879,24

ROVER,3,530,-1,-16.736,49.968,1.990,-272056.004,8146861.191,-4.589749343,-4.644023705,
137.442475987,-4503.030,2168.02,20132.76,542.93,5033.70,0.250,-1.391,45.225,399803902,26

SITE,4,-1,-1,-38.781,74.736,2.490,-272078.049,8146885.959,-4.590121255,-4.644399977,
137.442893838,-4503.530,2256.20,20231.84,564.97,5058.47,0.000,-0.000,0.000,400069973,29

ROVER,4,0,4,-38.781,74.736,2.490,-272078.049,8146885.959,-4.590121255,-4.644399977,
137.442893838,-4503.530,2256.20,20231.84,564.97,5058.47,-2.318,2.844,-120.443,400069973,29
```

```
ROVER,4,404,-1,-46.027,98.906,3.580,-272085.295,8146910.129,-4.590243500,-4.644523654,
137.443301601,-4504.620,2285.18,20328.52,572.22,5082.64,-3.112,-0.271,90.570,400867743,38
```

In reality the two lines for each entry above are on a single line with no spaces. Each column means the following:

`frame`: Frame type, SITE or ROVER.

`site`: Site number from the RMC.

`drive`: Drive number from the RMC. Will be -1 for SITE entries.

`pose`: Pose number from the RMC. Will be -1 for SITE entries, and for any view other than telemetry. Exception: for localized_pos only, an extra entry is added for ROVER(*n*,0) to show the rover orientation at the location where a Site frame was declared, as a convenience (this entry does not exist in the database, since the Site is localized, not Drive=0). This entry will have the highest pose number available for that Site *n* Drive 0, which is the one actually used for the entry.

`landing_x`: X coordinate relative to the landing site (SITE 1). +X points north.

`landing_y`: Y coordinate relative to the landing site (SITE 1). +Y points east.

`landing_z`: Z coordinate relative to the landing site (SITE 1). +Z points ***down***.

`northing`: Meters north of the equator (or south, if negative). Same as the X coordinate, in PLACES parlance. Most cartographers would call this +Y.

`easting`: Meters east of the prime meridian, as measured at the equator (see Section 3.9.1). Same as the Y coordinate in PLACES parlance. Most cartographers would call this +X.

`planetocentric_latitude`: Latitude of the point, measured using a planetocentric system. Planetocentric coordinates are measured as angles from the center of the planet. Latitude is computed from northing (X) using the formula:

$$\phi_{pc} = \frac{x}{R_e} \bullet \frac{180}{\pi}$$

where $R_e$ is the ellipsoid radius, or 3396190 meters.

`planetodetic_latitude`: Latitude of the point, measured using a planetodetic system. Planetodetic coordinates are measured as the tangent to the ellipsoidal surface at that point. Planetodetic latitude is computed using the formula:

$$\phi_{pd} = \tan^{-1}\left(\left(\tan\phi_{pc}\right)\left(\tfrac{R_e}{R_s}\right)^2\right)$$

where $\phi_{pc}$ is the planetocentric latitude in degrees, $R_e$ is the radius of the ellipsoid (3396190), $R_s$ is the radius of the sphere (3376200), and the trig functions are in degrees.

`longitude`: Longitude of the point. Longitude is computed from easting (Y) using the formula:

$$\theta = \frac{y}{R_e} \bullet \frac{180}{\pi}$$

where $R_e$ is the ellipsoid radius, or 3396190 meters. Note that there is no difference in longitude between the planetocentric and planetodetic systems, since Mars is modeled as an oblate spheroid, not an ellipsoid.

`elevation`: Elevation of the point relative to the datum. This uses the cartographic definition of elevation, so positive elevation is ***up***. It is thus computed as the negative of the global-frame Z value. For a treatment on converting elevation to radius, see Section 3.9.3.

`map_pixel_line`: 0-based line number of the point in the base map cutout image (see Section 4.5). Computed from northing (global-frame x) using the formula:

$$l_{map} = -\left(\frac{x - min\_x}{x\_scale}\right)$$

where min_x and x_scale come from the file `maps/orbital_map.lbl` in the `X_AXIS_MINIMUM` and `MAP_SCALE` (first value) keywords, respectively .

`map_pixel_sample`: 0-based sample number of the point in the base map cutout image (see Section 4.5). Computed from easting (global-frame y) using the formula:

$$s_{map} = -\left(\frac{y - \min\_y}{y\_scale}\right)$$

where min_y and y_scale come from the file `maps/orbital_map.lbl` in the `Y_AXIS_MINIMUM` and `MAP_SCALE` (second value) keywords, respectively.

`dem_pixel_line:` 0-based line number of the point in the DEM cutout image (see Section 4.5). Computed the same as `map_pixel_line` except the file `orbital_dem.lbl` is used as the source of the values.

`dem_pixel_sample:` 0-based sample number of the point in the DEM cutout image (see Section 4.5). Computed the same as `map_pixel_sample` except the file `orbital_dem.lbl` is used as the source of the values.

`roll`: Specifies the roll value of the rover at this location. Computed by decomposing the orientation quaternion into Euler angles. Roll is a clockwise rotation around the Rover frame +X axis (positive means left side is up). Data range -180 to +180 degrees.

`pitch`: Specifies the pitch value of the rover at this location. Computed by decomposing the orientation quaternion into Euler angles. Pitch is a clockwise rotation around the Rover frame +Y axis (positive means nose is up). Data range -90 to +90 degrees.

`yaw`: Specifies yaw value (or heading) of the rover at this location. Computed by decomposing the orientation quaternion into Euler angles. Yaw is a clockwise rotation about the Rover frame +Z axis. Since +Z points down, yaw is equivalent to heading, with north at 0 degrees, east at 90, south at 180, and west at 270. Data range from 0 to +360 degrees.

`sclk`: Earliest spacecraft clock (sclk) seen for the site/drive on this line. This can be derived via a lookup in the sclk table (Section 4.8.3) but is repeated here for convenience. Pose is ignored when doing this lookup. Fractional values are truncated down to integer.

`sol`: Solar day (sol) number associated with the sclk. This value is derived from the sclk using the SPICE "chronos" program.

Note that each .csv file has an associated .lbl file, which describes the format of the CSV file according to PDS standards. The .lbl file is not further described here.

### *4.5. Maps*

The `DATA/MAPS` directory contains a cutout of the "base map" that covers the area traversed by the rover during the time represented by this delivery. This base map is the reference used for all localizations by the MSL Project. It is important to realize that the localized positions, including latitude/longitude in the CSV file, are created using this map. So if the latitude/longitude are transferred to another map, the registration will only be as good as the coregistration of both maps. That is the primary reason the map is being included: so users can unambiguously identify localized positions by features, and decide for themselves how good the coregistration is between the MSL base map and the map they want to use.

The source base map is a 280,000 line by 180,000 sample mosaic at 0.25 m/pixel resolution covering Peace Vallis and the Peace Vallis fan, Aeolis Palus within the landing ellipse, and lower Aeolis Mons (aka Mt. Sharp), which covers the entire MSL operational area in Gale crater. It was made by the MSL Localization Team using mostly MRO HiRISE orthophoto data generated by the USGS, with holes filled by MRO CTX low-nadir angle map projected images, and registered to the Mars Express HRSC 50m/pixel DEM which is in turn registered to the MOLA gridded DEM.

The cutout is determined by looking at the minimum and maximum extents of the "localized_pos" view for the duration covered by this delivery, plus ~500 meters on all sides.

Additionally, `DATA/MAPS` contains a similar cutout of the Digital Elevation Model (DEM) map. This DEM contains the elevation (in meters) for every point in the map. The DEM is a 57,440 line by 32,980 sample image at 1.0 m/pixel resolution covering the MSL operational area (but not exactly the same coverage as the base map). It was also made by the MSL Localization Team using USGS MRO HiRISE DEMs, backfilled with a single CTX DEM created at JPL using Socet Set, and is similarly coregistered using the associated orthophotos.

Note however that the pixel grids of the DEM and base map do not align. The base map upper left corner is at an integer meter boundary (see the `msl_orbital_map.lbl` file for details). However, the DEM's upper left corner is not an integer. This 0.07519 x 0.49279 meter offset means the DEM and base map cutouts included in this delivery do not exactly coregister, and in fact the DEM cutout is one meter (one pixel) larger than you would expect based on the size of the base map cutout, in order to cover the extra fractional meters. In practice this makes very little difference; for most purposes you can treat the two as coregistered with little loss of accuracy. For systems that understand fractional pixel locations (such as most GIS programs), using the supplied minimum and maximum values from the label will give precise results. We decided reinterpolating the DEM to be 'pixel perfect' would introduce additional errors for no appreciable gain.

There are four files in the `DATA/MAPS` directory, described below.

### 4.5.1. msl_orbital_map.img, msl_orbital_map.lbl

The `msl_orbital_map.img` file contains the actual map. It is a standard byte-format dual-label VICAR/ODL image as described in the camera SIS [ 1 ]. The `msl_orbital_map.lbl` file is the detached PDS label for the image. The image is compliant with the camera SIS with the following exceptions:

- The label is minimal, containing only map extents and scale in addition to the standard image description keywords (see example label below). All keywords are compliant with the SIS except as noted below.
- MAP_PROJECTION_TYPE is EQUIRECTANGULAR, which is what the MSL localization team uses. The rectangular projection is characterized by parallel lines of latitude and longitude, with constant meters/pixel scale. See Section 3.9.1. Note that the x/y axis minimum and maximum values specify the corner coordinates in global northing (x) and easting (y) frame, and refer to the outside edges of the respective pixels. See the formulas in Section 4.4.2.4 for converting these to latitude/longitude.
- REFERENCE_COORD_SYSTEM_NAME is ORBITAL (index of 0), which is the global northing (x)/easting (y) frame. See Section 3.9.1.
- CENTER_LATITUDE (planetocentric) and CENTER_LONGITUDE have been added for convenience. However, the corner points should be considered the definitive location reference.
- The IMAGE_MAP_PROJECTION object has been added to facilitate using the map images in tools such as GDAL (http://gdal.org). The additional keywords in this object (which are standard PDS keywords) are listed and briefly described below in the context of this product only. The IMAGE_MAP_PROJECTION object is more fully defined in Appendix B.14 of the PDS Standards Reference [ 6 ], with additional description in the DATA_SET_MAP_PROJECTION catalog object in Appendex B.8.
    - MAP_PROJECTION_TYPE: declares the projection as EQUIRECTANGULAR.

- MAP_SCALE: the pixel size; pixels are square in this case.
- A_AXIS_RADIUS, B_AXIS_RADIUS, C_AXIS_RADIUS: sets the Mars spherical radius to 3393190 m in the datum; All three are the same.
- COORDINATE_SYSTEM_NAME: sets the latitude type to PLANETOCENTRIC.
- LINE_PROJECTION_OFFSET: the offset in the line direction from the projection origin (latitude=0.0) to the image upper left corner in pixels (i.e. the image coordinate of latitude=0.0). Values are positive to the right and down (thus they are inverted from what one might expect, since northing is positive up).
- SAMPLE_PROJECTION_OFFSET: the offset in the sample direction from the projection origin (longitude=0.0) to the image upper left corner in pixels (i.e. the image coordinate of longitude=0.0). Values are positive to the right and down.
- ^DATA_SET_MAP_PROJECTION: Points to a file in the CATALOG directory describing the map projection.
- CENTER_LATITUDE, CENTER_LONGITUDE: Defines the origin for the equirectangular projection. Always 0,0 in these products.
- COORDINATE_SYSTEM_TYPE: Constant “BODY-FIXED ROTATING”.
- LINE_FIRST_PIXEL, SAMPLE_FIRST_PIXEL: Always 1,1.
- LINE_LAST_PIXEL, SAMPLE_LAST_IPXEL: Same as LINES and LINE_SAMPLES in the IMAGE object.
- MAP_PROJECTION_ROTATION: Always 0.0.
- MAP_RESOLUTION: Resolution of map in pixels/degree. Computed as:
  $$\frac{2\pi\left(\mathrm{A\_AXIS\_RADIUS}\times 1000\right)}{360\times \mathrm{MAP\_SCALE}}$$
- POSITIVE_LONGITUDE_DIRECTION: Constant “EAST”.
- MINIMUM_LATITUDE, MAXIMUM_LATITUDE: Computed by converting X_AXIS_MINIMUM and X_AXIS_MAXIMUM to planetocentric latitude (see Section 4.4.2.4).
- EASTERNMOST_LONGITUDE, WESTERNMOST_LONGITUDE: Computed by converting Y_AXIS_MINIMUM and Y_AXIS_MAXIMUM to longitude (see Section 4.4.2.4).

An example of the detached PDS label is below:

```
PDS_VERSION_ID                   = PDS3
```

```
/* FILE DATA ELEMENTS */

RECORD_TYPE                        = FIXED_LENGTH
RECORD_BYTES                       = 23800
FILE_RECORDS                       = 24722
LABEL_RECORDS                      = 1

/* POINTERS TO DATA OBJECTS */

^IMAGE_HEADER                 = ("msl_orbital_map.img",2)
^IMAGE                        = ("msl_orbital_map.img",3)
OBJECT                             = IMAGE
  INTERCHANGE_FORMAT               = BINARY
  LINES                            = 24720
  LINE_SAMPLES                     = 23800
  SAMPLE_TYPE                      = UNSIGNED_INTEGER
  SAMPLE_BITS                      = 8
  BANDS                            = 1
  BAND_STORAGE_TYPE                = BAND_SEQUENTIAL
END_OBJECT                         = IMAGE

/* IDENTIFICATION DATA ELEMENTS */

DATA_SET_ID                        = "MSL-M-ROVER-6-RDR-PLACES-V1.0"
INSTRUMENT_HOST_NAME               = "MARS SCIENCE LABORATORY"
INSTRUMENT_NAME                    = "N/A"
PRODUCT_CREATION_TIME              = 2016-03-03T22:12:02
PRODUCT_ID                         = "msl_orbital_map"
PRODUCT_VERSION_ID                 = "1.0"
TARGET_NAME                        = "MARS"
MISSION_PHASE_NAME                 = "PRIMARY SURFACE MISSION"
SPACECRAFT_CLOCK_START_COUNT       = "397502188.872"
SPACECRAFT_CLOCK_STOP_COUNT        = "501099590.646337"
START_TIME                         = 2012-08-06T05:18:39
STOP_TIME                          = 2015-11-18T06:38:23
DESCRIPTION                        = "Data from base map derived from MRO
                                      HiRISE and CTX.  See SIS for details"

/* SURFACE PROJECTION DATA ELEMENTS */

GROUP                              = SURFACE_PROJECTION_PARMS
  MAP_PROJECTION_TYPE              = EQUIRECTANGULAR
  MAP_SCALE                        = (0.25 <m/pixel>,0.25 <m/pixel>)
  REFERENCE_COORD_SYSTEM_NAME      = "ORBITAL"
  REFERENCE_COORD_SYSTEM_INDEX     = 0
  X_AXIS_MINIMUM                   = -277694.0
  X_AXIS_MAXIMUM                   = -271514.0
  Y_AXIS_MINIMUM                   = 8141828.0
  Y_AXIS_MAXIMUM                   = 8147778.0
  CENTER_LATITUDE                  = -4.632735576
  CENTER_LONGITUDE                 = 137.40775306
END_GROUP                          = SURFACE_PROJECTION_PARMS

/* IMAGE MAP PROJECTION DATA ELEMENTS */
```

```
OBJECT                              = IMAGE_MAP_PROJECTION
  A_AXIS_RADIUS                     = 3396.19 <km>
  COORDINATE_SYSTEM_NAME            = "PLANETOCENTRIC"
  MAP_PROJECTION_TYPE               = EQUIRECTANGULAR
  LINE_PROJECTION_OFFSET            = -1086055.5
  SAMPLE_PROJECTION_OFFSET          = -32567311.5
  MAP_SCALE                         = 0.25 <m/pixel>
  ^DATA_SET_MAP_PROJECTION          = "DSMAP.CAT"
  B_AXIS_RADIUS                     = 3396.19 <km>
  C_AXIS_RADIUS                     = 3396.19 <km>
  CENTER_LATITUDE                   = 0.0
  CENTER_LONGITUDE                  = 0.0
  COORDINATE_SYSTEM_TYPE            = "BODY-FIXED ROTATING"
  LINE_FIRST_PIXEL                  = 1
  LINE_LAST_PIXEL                   = 24720
  MAP_PROJECTION_ROTATION           = 0.0
  MAP_RESOLUTION                    = 237098.7900932249 <pixel/degree>
  POSITIVE_LONGITUDE_DIRECTION      = EAST
  SAMPLE_FIRST_PIXEL                = 1
  SAMPLE_LAST_PIXEL                 = 23800
  MINIMUM_LATITUDE                  = -4.684865745469447
  MAXIMUM_LATITUDE                  = -4.580605407446296
  WESTERNMOST_LONGITUDE             = 137.3575630107385
  EASTERNMOST_LONGITUDE             = 137.4579431096443
END_OBJECT                          = IMAGE_MAP_PROJECTION

/* VICAR IMAGE HEADER DATA ELEMENTS */

OBJECT                              = IMAGE_HEADER
 HEADER_TYPE                        = VICAR2
 INTERCHANGE_FORMAT                 = ASCII
 BYTES                              = 23800
 ^DESCRIPTION                       = "VICAR2.TXT"
END_OBJECT                          = IMAGE_HEADER

/* ODL IMAGE HEADER DATA ELEMENTS */

OBJECT                              = ODL_HEADER
 HEADER_TYPE                        = ODL
 INTERCHANGE_FORMAT                 = ASCII
 BYTES                              = 23800
 ^DESCRIPTION                       = "ODL.TXT"
END_OBJECT                          = ODL_HEADER
END
```

Note that the actual map extents and image sizes will differ from the example above, depending on the size of the cutout needed for the specific PLACES delivery. Also note that the label refers to MSL rather than MRO even though the map is originally derived from MRO data. This is because the images have been vertically and horizontally georeferenced by the MSL Localization Team to become the base map, and the cutout provided here is specifically sized to match the extent of the MSL data contained in PLACES.

### 4.5.2. msl_orbital_dem.img, msl_orbital_dem.lbl

These files contain the DEM cutout. They are identical in format to the `msl_orbital_map.*` files, except that the image is in 32-bit floating point format. Note that the x/y extents are slightly different due to the fact that the pixel grids are not aligned, as described above.

### 4.5.3. Using localization data in GIS programs

The orbital datasets referenced in section 4.5.2 can be used directly in most GIS programs (e.g. ESRI ArcGIS) by treating the .lbl file as a spatial layer. To work correctly, the .lbl and .img file it references need to be in the same directory. All the pertinent image and projection attributes are retained in the .lbl file, hence making it natively readable by most mapping software. In ArcGIS version 10.3.1, there appears to be a bug that offsets the extents of both orbital datasets by less than a pixel (~0.125 m in easting for the map and less than 1 cm in the DEM); the user may wish to either correct the offset after importing into ArcGIS or use another program, like GDAL, to convert the file first to maintain the correct extent (as of this writing, GDAL does not exhibit the same bug)

Below is an example using the GDAL program gdal_translate (version 1.11.0) to convert the default PDS format to a GeoTIFF with the correct extent:

```
gdal_translate msl_orbital_map.lbl msl_orbital_map.tif

gdal_translate msl_orbital_dem.lbl msl_orbital_dem.tif
```

The GeoTIFFs above have the correct extent in ArcGIS.

## *4.6. Orbital*

The `EXTRAS/XML_ANCILLARY_DATA/orbital` directory contains information on all the ORBITAL frames defined in PLACES. Orbital frames generally correspond to a map image but can also be defined for other reasons. For example, orbital(1) is the base map (see Section 4.5.1) while orbital(2) is the DEM that goes along with it (it's a different frame because the resolutions differ). Orbital(601) is one specific HiRISE orthophoto (not a mosaic) that has been coregistered with the base map. Note that the index number (other than 0, which is the global frame) is completely arbitrary and is whatever was assigned during operations. They have no relationship to rover Site frames.

Unlike rover locations, the location of an orbital map frame is known when it is created. Therefore, they do not generally have multiple solutions. Although multiple solutions are allowed by the database, they have not been used so they are not reflected in the orbital data for PDS.

There are three files for each orbital frame, and one summary file, all described below.

### 4.6.1. all_orbitals file

The `all_orbitals.xml` file contains a list of all orbital frames included in this delivery. An example is shown below:

```
<list immutable="true">
    <item date_added="2012-08-02 02:00:45.0" site="0" frame="ORBITAL"/>
```

```
    <item date_added="2012-08-02 02:00:45.0" site="1" frame="ORBITAL"/>
    <item date_added="2012-08-02 02:00:45.0" site="2" frame="ORBITAL"/>
    <item date_added="2014-04-04 22:33:59.0" site="3" frame="ORBITAL"/>
    <item date_added="2014-04-07 17:46:23.0" site="4" frame="ORBITAL"/>
    <item date_added="2012-08-05 08:32:11.0" site="101" frame="ORBITAL"/>
...
</list>
```

Each `<item>` tag contains the date the orbital frame was added, the frame number (under "`site`"), and the constant frame name "`ORBITAL`". Note that there are gaps in the numbering; orbital frame numbers are chosen for operational convenience and to group related frames together.

### 4.6.2. Metadata

The `orbital_`***`n`***`_metadata.xml` file contains keyword-value pairs of metadata that were entered when the orbital frame was created. There is no formal standardization of the keywords (any keyword/value pair can be added by the user during operations), but the common ones are described below.

An example of this file (for orbital(1)) follows:

```
<metadata>
    <item>
        <key>lower_right_northing_m</key>
        <value>-311164</value>
    </item>
    <item>
        <key>pixel_definitions</key>
        <value>Line/samp of (0,0) is the upper-left corner of the upper-left pix
el</value>
    </item>
    <item>
        <key>upper_left_easting_m</key>
        <value>8126248</value>
    </item>
    <item>
        <key>coord_sys_definition</key>
        <value>+X is North, +Y is East, +Z is Down</value>
    </item>
    <item>
        <key>z_scale</key>
        <value>1.0</value>
    </item>
    <item>
        <key>center_latitude</key>
        <value>0</value>
    </item>
    <item>
        <key>lower_right_easting_m</key>
        <value>8171248</value>
    </item>
    <item>
        <key>projection</key>
        <value>Equirectangular</value>
```

```
</item>
<item>
    <key>ellipsoid_radius</key>
    <value>3396190</value>
</item>
<item>
    <key>pixel_type</key>
    <value>unsigned_integer</value>
</item>
<item>
    <key>latitude_type</key>
    <value>planetocentric</value>
</item>
<item>
    <key>origin_sample</key>
    <value>0</value>
</item>
<item>
    <key>origin_line</key>
    <value>0</value>
</item>
<item>
    <key>upper_left_northing_m</key>
    <value>-241164</value>
</item>
<item>
    <key>samples</key>
    <value>180000</value>
</item>
<item>
    <key>x_scale</key>
    <value>0.25</value>
</item>
<item>
    <key>description</key>
    <value>25cm MSLICE basemap</value>
</item>
<item>
    <key>filename</key>
    <value>MSL_Gale_MSLICE_HIRISE_Mosaic_25cm.tif</value>
</item>
<item>
    <key>y_scale</key>
    <value>0.25</value>
</item>
<item>
    <key>lines</key>
    <value>280000</value>
</item>
<item>
    <key>pixel_depth</key>
    <value>8</value>
</item>
<item>
    <key>projection_units</key>
```

```
        <value>meters</value>
    </item>
    <item>
        <key>center_longitude</key>
        <value>0</value>
    </item>
    <item>
        <key>formula</key>
        <value>line = line_origin - x/x_scale; samp = samp_origin + y/y_scale;</
value>
    </item>
</metadata>
```

Some of the common keywords (in `<key>`) are described below. As mentioned above, the set of keywords is not standardized so there may be others not described here, and not all these will be present in all frames.

`center_latitude`: Latitude of the center of the projection. Always 0 for the global frame.

`center_longitude`: Longitude of the center of the projection. Always 0 for the global frame.

`coord_sys_definition`: A text description of the coordinate system orientation.

`description`: A text description of the frame.

`ellipsoid_radius`: Radius of the Mars ellipsoid (in meters) used for latitude/longitude conversion.

`filename`: Filename of the (primary) image described by this frame. This file is not necessarily available in PDS.

`formula`: Describes how to use the values.

`latitude_type`: Planetocentric or planetodetic.

`lines`: Number of lines in the image.

`lower_right_easting_m`: Easting (in meters, see Section 3.9.1) of the lower right corner.

`lower_right_northing_m`: Northing of the lower right corner.

`origin_line`: Line number in the image of the origin (i.e. the point defined by the `orbital_`***n***`_origin.xml` file). Lines start counting at 0 in this context (see pixel_definitions).

`origin_sample`: Sample number in the image of the origin (i.e. the point defined by the `orbital_`***n***`_origin.xml` file). Samples start counting at 0 in this context (see pixel_definitions).

`pixel_definitions`: Describes how pixels are defined.

`pixel_depth`: Number of bits in a pixel.

`pixel_type`: Data type of the pixel.

`projection`: Projection for the image. Should be "Equirectangular" in all cases.

`projection_units`: Units used in the projection. Should be "meters" in all cases.

`samples`: Number of samples in the image.

`upper_left_easting_m`: Easting (in meters, see Section 3.9.1) of the upper left corner.

`upper_left_northing_m`: Northing of the upper left corner.

`x_scale`: X scale factor in meters per pixel.

`y_scale`: Y scale factor in meters per pixel.

`z_scale`: Z scale factor. Should be 1.0 since all DEM data are in meters.

#### 4.6.3. Origin

The `orbital_`***n***`_origin.xml` file specifies where the origin of the map image is in relation to the reference frame with which it was created. In other words, the reference frame is whatever was entered by the user when the frame was created (contrast with the next section). This preserves the original data, before any coordinate transforms have been applied.

The format is the same as the localization raw files described in Section 4.4.2.2, so the description is not repeated here. The important values are the offset and the reference frame for the offset. The offset is the location of the point defined by origin_line and origin_sample in the metadata.

An example file (for orbital(1)) is below.


```
<solution immutable="true" add_date="" solution_id="telemetry">
    <rmc site="1" frame="ORBITAL"/>
    <offset z="0.0" y="8126248.0" x="-241164.0">
        <solution_metadata>
            <derived_from date_added="2012-08-02 02:00:45.0" site="0"
frame="ORBITAL"/>
            <reference_rmc date_added="2012-08-02 02:00:45.0" site="0"
frame="ORBITAL"/>
            <reference_view>telemetry</reference_view>
        </solution_metadata>
    </offset>
    <orientation v3="0.0" v2="0.0" v1="0.0" s="1.0">
        <solution_metadata>
            <derived_from date_added="2012-08-02 02:00:45.0" site="0"
frame="ORBITAL"/>
            <reference_rmc date_added="2012-08-02 02:00:45.0" site="0"
frame="ORBITAL"/>
            <reference_view>telemetry</reference_view>
        </solution_metadata>
    </orientation>
    <origination institution="JPL" creation_date="2012-08-02" add_date="2012-08-
```

```
02" method="init script" user="hbmorten"/>
</solution>
```

### 4.6.4. Global Origin

The `orbital_n_origin_global.xml` file specifies where the origin of the map image is in relation to the global coordinate system. In other words, the reference frame is orbital(0), regardless of what was entered by the user when the frame was created (contrast with the previous section). The common origin makes this file more generally useful than the (non-global) origin file.

The format is the same as the global-frame files described in Section 4.4.2.3, so the description is not repeated here. The important value is the offset. The offset is the location of the point defined by origin_line and origin_sample in the metadata.

An example file (for orbital(1)) is below.

```
<translate immutable="false">
    <view>telemetry</view>
    <reference-rmc date_added="2012-08-02 02:00:45.0" site="0" frame="ORBITAL"/>
    <offset z="0.0" y="8126248.0" x="-241164.0"/>
</translate>
```

## *4.7. Mechanisms*

The primary purpose of PLACES is to store localization data. However, it also stores mechanism information – mostly joint angles for articulated mechanisms. Unlike localizations, there are no solutions for mechanisms. The mechanism values are assumed to be known, with no room for differing opinions. While this is not always the case in reality (for example, mast pointing is often adjusted during mosaic processing), such updates are not captured in PLACES.

Each mechanism's data is contained in a single file in the "`EXTRAS/XML_ANCILLARY_DATA/mechanisms`" directory. These files are:

- `mech_arm.xml`
- `mech_chimra.xml`
- `mech_drill.xml`
- `mech_drive.xml`
- `mech_drt.xml`
- `mech_hga.xml`
- `mech_ic.xml`
- `mech_rsm.xml`
- `mech_suspension.xml`

### 4.7.1. Sources of Information

Mechanism information is collected from various places, depending on the mechanism. Each one's source data are described in the appropriate section. However, there are two sources of information that are common to most mechanisms; they are described below.

This document is not intended to document the information sources in detail. The information is provided in order to better understand the data, for those who are familiar with the telemetry data. Understanding the data source is not critical to using these data.

#### 4.7.1.1. Telemetry Common Header

Every MSL data product sent via the telemetry mechanism has a header associated with it. This common header is contained in the "`.emd`" file in telemetry, and contains a snapshot of the rover state at the time the data product was created. PLACES monitors the header for all telemetry products (with a few exceptions for those with known problems) and enters all the information into PLACES.

It should be noted that rover locations and orientations (for the "`telemetry`" view) are also extracted from this common header.

#### 4.7.1.2. Image Labels

Every image product (engineering cameras, MMM cameras, and RMI) contains metadata labels, as described in the Camera SIS [ 1 ]. These metadata contains a snapshot of the rover state at the time the image was taken (which is not necessarily the same as the time the data product was created, thus the need to look at both the image labels and the telemetry common header). The available data in this snapshot varies per instrument and mode; see the SIS for details.

PLACES monitors all images and ingests the information it finds. Rover locations and orientations (for the "telemetry" view) are also extracted from this common header.

#### 4.7.1.3. Specific Data Products

Other mechanisms get information by monitoring data products specific to the mechanism, or a general rover kinematics state product (called `MotRks`). These are listed in the individual mechanisms' descriptions.

### 4.7.2. Common Format

All mechanisms have a common format. For example, the arm mechanism file looks like this:

```
<mechanisms>
    <mechanism>
        <rmc approx_min_sclk_printable="400696851"
approx_max_sclk_printable="405128150" min_sclk_printable="400696851"
max_sclk_printable="405128150" arm="0" drive="0" site="0" frame="INMOTION"/>
        <arm enc_wrist="0.0" enc_turret="0.0" enc_elevation="0.0" enc_elbow="0.0
" enc_azimuth="0.0"/>
    </mechanism>
    <mechanism>
        <rmc approx_min_sclk_printable="397502188.872"
approx_max_sclk_printable="397504943" min_sclk_printable="397502188.872"
max_sclk_printable="397504943" arm="0" drive="0" site="1" frame="INMOTION"/>
        <arm enc_wrist="0.0" enc_turret="0.0" enc_elevation="0.0"
enc_elbow="0.0" enc_azimuth="0.0"/>
    </mechanism>
...
```

```
</mechanisms>
```

Within the umbrella `<mechanisms>` element are some number of `<mechanism>` elements. Each one describes the state of the mechanism at one specific point in time, described by the `<rmc>` element. The actual mechanism element (`<arm>` in this example) varies per mechanism, as described in each section.

The `<rmc>` element describes when the mechanism information is valid. Its elements are described below:

`approx_min_sclk_printable`: Printable version of the approximate minimum sclk.

`approx_max_sclk_printable`: Printable version of the approximate maximum sclk.

`min_sclk_printable`: Printable version of the minimum sclk.

`max_sclk_printable`: Printable version of the maximum sclk.

`frame`: Always INMOTION for mechanism files, should be ignored.

`site, drive, pose, arm, chimra, drill, rsm, hga, drt, ic`: RMC components at the time this entry was valid. In general, mechanism files will only have site, drive, and the RMC component relevant for the mechanism in question.

In the above, the min and max sclk define the exact range over which it is known that the mechanism had this value. If they are the same, the mechanism could have been in motion, with that value for only an instant in time (the corresponding RMC element should be odd if it was in motion). Or it could be that PLACES had no information other than a specific instant in time. If the values are not the same, then it was known that the mechanism was in the same position for the entire time period. It is possible for the min and max sclk values to be null ("") or missing, if no exact values are known.

The approximate sclk's exist because one of the major sources of information (telemetry common header) reports sclk values only as integer seconds. Subseconds are not available. Thus, the approximate sclk's are often rounded to whole integers. Unlike the non-approximated values, approximated values will always exist (they will often be identical to the non-approximated values, if the telemetry common header added no information).

The `_printable` suffix indicates the sclk is reported as a standard decimal number. The database output includes attributes without `_printable`, which are in scientific notation. For clarity, these elements are removed from the PDS output, since they add no additional information.

#### 4.7.2.1. Mechanism Values and Meanings

Note that within a given mechanism element, not all possible attributes will be present, depending on what was available in the source data. Generally these attributes will simply be absent, although a null ("") value is possible. Internally, values of `1.0e+30` flag missing values, but these should not appear in the PDS output.

All mechanism values are in radians unless specified otherwise.

It is generally beyond the scope of this document to describe the exact meaning of these mechanism values – what exactly is being described, what 0 means, and what rotation direction is positive. This information should be available in the PPPCS ([ 3 ], [ 4 ]), or in other MSL project documentation. Note that the PPPCS Volume 9 [ 4 ], which defines in detail the coordinate frames and kinematics for the RSM and mobility systems, is included in the PLACES PDS delivery under the DOCUMENTS directory.

#### 4.7.2.2. Resolvers vs. Encoders

Several of the mechanism angles are measured by two different sensors. The "encoder" is on the input side of the shaft, attached to the motor. The "resolver" is on the output side of the shaft, attached to the moving mechanism. Thus the encoder records motor movements, while the resolver records how the device actually moved (including the effects of backlash). For most purposes, the resolver is more accurate (since it takes backlash into account) and should be used if available. It is possible for the resolver data to be unavailable (say, in the event of hardware failure), in which case the encoder should be used instead.

### 4.7.3. Arm

The state of the rover's robotic arm is described in the `mech_arm.xml` file. The `<arm>` element can contain the following attributes:

- `enc_azimuth`
- `enc_elevation`
- `enc_elbow`
- `enc_wrist`
- `enc_turret`
- `res_azimuth`
- `res_elevation`
- `res_elbow`
- `res_wrist`
- `res_turret`

The five joints are numbered, from the rover out, in the order (azimuth, elevation, elbow, wrist, turret). Both encoder and resolver values are available.

In addition to the telemetry header and image labels, arm information comes from the `MotRks` and `ArmSummary` telemetry products.

### 4.7.4. CHIMRA

The state of the CHIMRA (Collection and Handling for Interior Martian Rock Analysis) mechanism on the robotic arm turret is described in the `mech_chimra.xml` file. The `<chimra>` element can contain the following attributes:

- `portioner`
- `scoop`
- `scoop_thwack`

- `thwack`
- `scoop_tcp_azimuth`
- `scoop_tcp_elevation`

CHIMRA data come only from the `MotRks` telemetry product.

### 4.7.5. Drill

The state of the drill mechanism on the robotic arm turret is described in the `mech_drill.xml` file. The `<drill>` element can contain the following attributes:

- `chuck`
- `feed`
- `rotate`

Drill data come only from the `MotRks` telemetry product.

### 4.7.6. Drive

The state of the rover wheels and steering is described in the `mech_drive.xml` file. The `<drive>` element can contain the following attributes:

- `drive_LF`
- `drive_LM`
- `drive_LR`
- `drive_RF`
- `drive_RM`
- `drive_RR`
- `steer_LF`
- `steer_LR`
- `steer_RF`
- `steer_RR`

The `drive_` attributes describe the rotation of the wheels. These are not yet implemented. The `steer_` attributes describe the steering of the wheels (and are implemented). The wheels are named by (Left|Right) (Front|Middle|Rear).

Drive data come from the `MobSummaryReport` and `MotRks` telemetry data products, in addition to the image labels.

### 4.7.7. DRT

The state of the DRT (Dust Removal Tool) mechanism on the robotic arm turret is described in the `mech_drt.xml` file. The `<drt>` element can contain the following attribute:

- `drt`

DRT data come only from the `MotRks` telemetry product.

### 4.7.8. HGA

The state of the HGA (High Gain Antenna) mechanism on the rover is described in the `mech_hga.xml` file. The `<hga>` element can contain the following attributes:

- `azimuth`
- `elevation`

HGA data come from the `HgaDpSummary` and `MotRks` telemetry data products, in addition to the image labels.

### 4.7.9. IC

The state of the IC (Inlet Cover) mechanism on the rover body is described in the `mech_ic.xml` file. The `<ic>` element can contain the following attributes:

- `sam_1`
- `sam_2`
- `chemin`

IC data come only from the `MotRks` telemetry product.

### 4.7.10. RSM

The state of the RSM (Remote Sensing Mast) mechanism (camera head) on the rover mast is described in the `mech_rsm.xml` file. The `<rsm>` element can contain the following attributes:

- `enc_azimuth`
- `enc_elevation`
- `res_azimuth`
- `res_elevation`

RSM azimuth and elevation are raw joint angles. For elevation, it is in the range of 0 to 182 degrees (but measured in radians). 0 is against the lower hardstop, which is pointing one degree past straight down, and positive is up. Pointing level to the Rover Nav frame is thus nominally at elevation 91 degrees. Azimuth is in the range 0 to 362 degrees (again measured in radians). 0 is at the leftmost hardstop (looking one degree past the back of the rover), increasing to the right. Pointing straight ahead in the Rover Nav frame is thus nominally at azimuth 181 degrees.

Calibration data show that the straight ahead and level position is actually at azimuth 3.167345 radians (181.4755 degrees), elevation 1.588171 radians (90.9955 degrees).

The above information is provided for convenience only, because of the importance of the RSM to many instrument observations. The PPPCS Volume 9 [ 4 ] should be considered the definitive reference, and any discrepancies should be resolved in favor of the PPPCS.

RSM data come from the `RsmSummary` and `MotRks` telemetry data products, in addition to the image labels and telemetry common header.

#### 4.7.11. Suspension

The state of the rover suspension is described in the `mech_suspension.xml` file. The `<suspension>` element can contain the following attributes:

- `bogie_left`
- `bogie_right`
- `diff_left`
- `diff_right`

where `bogie_` indicates the state of the rocker bogies, and `diff_` indicates the state of the differentials.

Suspension data come from the `MobSummaryReport` telemetry data product, in addition to the image labels.

### *4.8. Ancillary Tables*

There are several (currently, 3) ancillary tables, managed by PLACES, that provide additional information. These tables are in the "`EXTRAS/XML_ANCILLARY_DATA/ancillary`" directory and are:

- `observations.xml`
- `saved_frames.xml`
- `sclk_table.xml`

These are described individually below. Note that PLACES also provides an "instruments" table, but since that merely lists the constant instrument names, it is not included in PDS.

#### 4.8.1. Observations table

This is a table, in "`EXTRAS/XML_ANCILLARY_DATA/ancillary/observations.xml`" that lists exposure times for all of the camera observations seen by PLACES, referenced to the SCLK at which the exposure was taken. This table was created in response to a very specific request from a customer, and may or may not be generally useful. Note that MMM "recovered" products ("ERD" type) are not included in this table, and thus none of the descent MARDI images are included.

The table looks like this:

```
<observations>
    <image inst_id="MAHLI" sclk_str="400696851" duration="0.0"/>
    <image inst_id="MAHLI" sclk_str="400696854" duration="0.0"/>
    <image inst_id="MAST_LEFT" sclk_str="400703227" duration="2.6316"/>
    <image inst_id="MAST_RIGHT" sclk_str="400703227" duration="4.0541"/>
    <image inst_id="MAST_RIGHT" sclk_str="400703285" duration="4.0541"/>
...
</observations>
```

Within each `<image>` tag are the following attributes:

`inst_id`: Which instrument acquired the image.

`sclk_str`: Value of SCLK (spacecraft clock) when the image was taken. Note that the MMM cameras are precise only to integer seconds, while the engineering and Chemcam RMI cameras are precise to milliseconds.

`duration`: Duration of the exposure, in seconds.

### 4.8.2. Saved Frames Table

This is a table, in "`EXTRAS/XML_ANCILLARY_DATA/ancillary/saved_frames.xml`", that contains the mapping of saved frames to the site/drive/pose corresponding to them. Saved frames are a FSW concept, they are a shorthand or alias indicating a frame that the rover is remembering onboard, which can be used for commanding. Saved frames have no particular meaning on the ground; the RMC (site/drive/pose) triplet should be used instead (see also Section 2.4). This file contains a mapping between the two.

CAUTION: As of this writing, the saved frame table has not been validated and is likely to be unreliable. It is included for completeness, because it is stored in the database. However, there are known issues in how the data are being gathered.

An example is below. The fields should be self-explanatory.

```
<savedFrames><savedFrame frame_number="1" site="4" drive="0" pose="0" />
<savedFrame frame_number="2" site="4" drive="2100" pose="4" />
<savedFrame frame_number="3" site="4" drive="3232" pose="4" />
<savedFrame frame_number="4" site="4" drive="3474" pose="4" />
<savedFrame frame_number="5" site="5" drive="0" pose="0" />
...
</savedFrames>
```

### 4.8.3. SCLK Table

This is a table, in "`EXTRAS/XML_ANCILLARY_DATA/ancillary/sclk_table.xml`", that contains the SCLK (spacecraft clock) ranges for every RMC seen by PLACES. It is perhaps the most useful of the ancillary tables.

A "static" (or stable) RMC is one for which all (relevant) indices (omitting Site) are even (see Section 2.2). This indicates the rover is not moving. A dynamic RMC (where the index is odd) indicates that the rover, or some mechanism on the rover, was moving.

Both are stored in the SCLK table. The minimum and maximum SCLK's seen for that static segment, or that in-motion segment, are recorded.

For static RMC's, the time period indicates the period during which the rover was **known** to be static. It does ***not*** mean the rover stopped moving just before the time period or started again just after. It could very well have been quiescent for quite some time before or after the indicated times, and we simply don't have the data to tell. We just know it was not moving during that time period. Similarly, a range of times for a dynamic RMC means it was moving during that time, but not that it was static outside the time. Bottom line, there are gaps in the SCLK's during which we do not know, based on available telemetry, what the RMC was.

The SCLK table is pulled together from many different sources of information (basically, all the sources described in Section 4.7.1). Many of these data sources do not provide a full RMC. For example, the HGA motion report contains the site, drive, and HGA indices, but not the rest. So there are many entries in the SCLK table with -1 for various (mechanism, generally) RMC's. The given times apply just to the HGA motion; it is known for example that the HGA was static during the given interval if the HGA index is provided, but it is not known that the RSM was static if its index is -1. So each entry applies only to the RMC elements provided; it says nothing about when a mechanism was static if the corresponding RMC index is -1. This creates a lot of overlap in the ranges. For example the HGA might be known to be static for the entire duration of an arm activity.

Because the SCLK table comes from so many different sources, the possibility exists for small inconsistencies in the table, with overlapping SCLK ranges. These overlaps should generally be less than a second.

An example of the SCLK table, with several different kinds of entries, is provided below.


```
<list><item min="" max="" min_approx="400774612" max_approx="400774612" frame="R
OVER" site="4" drive="0" pose="2" arm="455" chimra="22" drill="38" rsm="652" hga
="1050" drt="0" ic="24"></item>
<item min="" max="" min_approx="400775545" max_approx="400775545" frame="ROVER"
site="4" drive="0" pose="2" arm="471" chimra="22" drill="38" rsm="652" hga="1050
" drt="0" ic="24"></item>
<item min="" max="" min_approx="400770813" max_approx="400770813" frame="ROVER"
site="4" drive="0" pose="2" arm="440" chimra="22" drill="38" rsm="648" hga="1033
" drt="0" ic="24"></item>
...
<item min="400770830.475045" max="400770830.475862"
min_approx="400770830.475045" max_approx="400770830.475862" frame="INMOTION"
site="4" drive="0" pose="-1" arm="-1" chimra="-1" drill="-1" rsm="-1" hga="1034"
drt=
"-1" ic="-1"></item>
<item min="400771245.389775" max="400771307.914072"
min_approx="400771245.389775" max_approx="400771307.914072" frame="INMOTION"
site="4" drive="0" pose="-1" arm="-1" chimra="-1" drill="-1" rsm="-1" hga="1038"
drt=
"-1" ic="-1"></item>
<item min="400771314.583499" max="400771397.914038"
min_approx="400771314.583499" max_approx="400771397.914038" frame="INMOTION"
site="4" drive="0" pose="-1" arm="-1" chimra="-1" drill="-1" rsm="-1" hga="1040"
drt=
"-1" ic="-1"></item>
...
<item min="448280042.913567" max="448280065.483718"
min_approx="448280042.913567" max_approx="448280065.483718" frame="INMOTION"
site="30" drive="484" pose="6" arm="0" chimra="0" drill="0" rsm="60" hga="0"
drt="0" ic="0"></item>
<item min="448280127.365807" max="448280155.483631"
min_approx="448280127.365807" max_approx="448280155.483631" frame="INMOTION"
```

```
site="30" drive="484" pose="6" arm="0" chimra="0" drill="0" rsm="62" hga="0"
drt="0" ic="0"></item>
<item min="448280164.975155" max="448280185.921173"
min_approx="448280164.975155" max_approx="448280185.921173" frame="INMOTION"
site="30" drive="484" pose="6" arm="0" chimra="0" drill="0" rsm="64" hga="0"
drt="0" ic="0"></item>
</list>
```


The attributes in the `<item>` element are:

`min`: Minimum sclk.
`max`: Maximum sclk.
`min_approx`: Approximate minimum sclk.
`max_approx`: Approximate maximum sclk.
`frame:` Internal PLACES value, should be ignored for PDS.
`site, drive, pose, arm, chimra, drill, rsm, hga, drt, ic`: RMC components for this sclk entry.

In the above, the min and max sclk define the exact range over which it is known that the RMC was valid. It is possible for the min and max sclk values to be null ("") or missing, if no exact values are known.

The approximate SCLKs exist because one of the major sources of information (telemetry common header) reports SCLK values only as integer seconds. Subseconds are not available. Thus, the approximate SCLKs are often rounded to whole integers. Unlike the non-approximated values, approximated values will always exist (they will often be identical to the non-approximated values, if the telemetry common header added no information).

# 5. Glossary

**areoid:** Shape defining "sea level" on Mars, taking into account local gravity variations. Same as geoid on Earth.

**attitude**: Synonymous with "orientation".

**chain**: A set of links between Derived-from frames that implement the system by which one localization can affect other downstream ones.

**clone**: A copy of a view; shares the same parent but has a copy of the contents.

**closed view**: A view that cannot be modified. Views can be closed up to a certain point, identified with an RMC value.

**coordinate frame**: As used in this document, a method by which Cartesian (XYZ) coordinates can be expressed. When used alone, refers to a type of frame, such as Rover or Site. With an index, refers to a specific instance of that frame type, valid at some point in time, with a specific origin and orientation (with respect to some other frame).

**data product**: A file sent by the rover containing raw telemetry information of a certain type.

**deep copy**: An expensive copy of a view, which may include several levels of parents.

**derived-from frame**: The frame from which a solution was derived. Need not be the same as the Reference frame.

**drive**: Second index of the RMC; indicates when the rover is driving (odd) or stationary (even).

**dynamic RMC**: An RMC with an odd value for Drive or Pose; indicates the rover is in motion or updating pose knowledge. Can also refer to odd values of mechanism RMC values, indicating the corresponding mechanism is in motion.

**easting**: Global-frame measurement of coordinates; indicates meters east of the prime meridian, as measured at the equator.

**estimate**: Synonymous with "solution".

**frame**: An instance of a coordinate system. Combines the type of frame (Site, Rover, Local_Level, Orbital) with an instance index (RMC, SCLK, or counter). The frame has a single but unknowable true pose; solutions provide estimates of that true pose.

**global frame**: A coordinate frame used for mapping, with its origin at the equator and prime meridian. Global frame coordinates are directly translatable to latitude/longitude. The global frame is an instance of an orbital frame, named by ORBITAL(0).

**groups**: A mechanism to control access to views and solution IDs.

**localization**: The process of determining where the rover is and how it is oriented at any given time.

**local level frame**: A combination of Rover frame position with Site frame orientation.

**mechanism**: Any device on the rover that can move, such as wheels, arms, covers, or masts.

**mechanism RMC**: The last 7 values of RMC (Arm, CHIMRA, Drill, RSM, HGA, DRT, IC), which indicate motions of mechanisms.

**metadata**: Descriptive information attached to database entries (views, solution ID's, etc) that plays no part in coordinate transformations.

**mobility RMC**: The first 3 values of RMC (Site, Drive, Pose), which have special meaning for mobility and the PLACES database.

**northing**: Global-frame measurement of coordinates; indicates meters north of the equator.

**open view**: A view that can be modified. See closed view.

**orientation**: How the rover or a frame is oriented in space. Specifically, a rotation relative to the reference frame.

**orbital frame**: Any of several coordinate frames describing maps or map-like entities, distinguished by an index. The global frame is an instance of an orbital frame (index 0).

**orbital images**: Maps used as an aid to localization and areoreferencing.

**PLACES**: Database storing localization solutions for all users across the MSL project.

**pose**: The combination of position and orientation of the rover or a frame. When capitalized, indicates the 3rd index of the RMC.

**position**: Where the rover or a frame is located. Specifically, a translation or offset in Cartesian XYZ space from the reference frame.

**quaternion**: A 4-element vector used to represent a rotation in 3-dimensional space. Used to represent orientation of the rover in PLACES.

**reference frame**: The frame in which coordinates (offsets or rotations) are expressed. See also Derived-from frame.

**ReST web service**: A mechanism for implementing web services using simple URL's and XML files.

**root**: The frame at the root of the coordinate frame tree, from which all others are derived. Defined to be the same as the global frame origin.

**rover (navigation) frame**: A coordinate frame attached to the rover, with +X pointing forward and +Z toward the belly, with +Y completing the right-handed system.

**RMC (Rover Motion Counter)**: A set of 10 indices used as a clock to track rover motion. Used as the instance identifier for most Rover frames. The first three are the mobility RMC, the rest are the mechanism RMC.

**saved frame**: A mechanism used onboard to save and reuse some frames after the rover moves from that location.

**site**: An area of operations; establishes a coordinate system origin for use in local operations.

**site frame**: A coordinate frame attached to the ground, with +X pointing north and +Z down according to the local gravity vector. Coincident with local_level at the time it is defined.

**sol**: One Martian day.

**solution**: One specific position or orientation (out of possibly many) for a given RMC. Represents an individual estimate of the unknown true value of position or orientation.

**solution ID**: A string used to identify a set of solutions.

SPICE: A toolkit to compute ancillary navigation-related data for space missions.

**stable RMC**: An RMC with even values for the Drive and Pose indices; indicates the rover is not intentionally moving or updating its pose knowledge. Can also apply to mechanism RMC's where even indicates the mechanism is not in motion.

**subframe**: A database implementation detail; combines dynamic RMC and unsolicited move frames in a single term.

**telemetry**: View containing only values obtained from the rover; it is the root of the view tree.

**unit vector**: A 3-element vector with magnitude 1.0, used to indicate a direction in space (such as the direction a camera is pointing) independent of position.

**unsolicited move**: An unexpected motion of the rover, such as slipping off a rock due to a wind gust.

**view**: A named entity which provides a single solution for each RMC. Used to differentiate different sets of solutions.

**visual odometry (visodom)**: A mechanism to refine pose onboard by analyzing images before and after a drive.

**XYZ**: Cartesian coordinates used to express an offset or position.

# 6. Acronym List

**CHIMRA**: Collection and Handling for Interior Martian Rock Analysis

**CSV**: Comma Separated Value

**CTX**: Context Camera (MRO)

**DEM**: Digital Elevation Model

**DRT**: Dust Removal Tool

**EVR**: Event Verification Record

**FSW**: Flight Software

GDAL: Geospatial Data Abstraction Library

**GIS**: Geographic Information System

GeoTIFF: Geographic Tagged Image File Format

**HiRISE**: High Resolution Imaging Science Experiment (on MRO)

**HGA**: High Gain Antenna

**HRSC**: High Resolution Stereo Camera (on Mars Express)

**HTTP**: HyperText Transfer Protocol

**Hz**: Hertz

**IC**: Inlet Cover

**ID**: Identifier

**IDPH**: Image (or Instrument) Data Product Header

**IMU**: Inertial Measurement Unit

**JPL**: Jet Propulsion Laboratory

**LDAP**: Lightweight Directory Access Protocol

**MAHLI**: Mars Hand Lens Imager

**MARDI**: Mars Descent Imager

**MER**: Mars Exploration Rover

**MIPL**: Multimission Image (or Instrument) Processing Laboratory

**MMM**: Mastcam, MAHLI, MARDI

**MOLA**: Mars Observer Laser Altimeter

**MRO**: Mars Reconnaissance Orbiter

**MSL**: Mars Science Laboratory

**OPGS**: Operations Product Generation Subsystem

**PDS**: Planetary Data System

**PLACES**: Position Localization and Attitude Correction Estimate Storage

PPPCS: Pointing, Positioning, Phasing, and Coordinate Systems (document)

**ReST**: Representational State Transfer

**RMC**: Rover Motion Counter

**RSM**: Remote Sensing Mast

**SCLK**: Spacecraft Clock

SPICE: Spacecraft, Planet, Instrument, Camera, Events

**UHF**: Ultra High Frequency

**URL**: Uniform Resource Locator

**USGS**: United States Geological Survey

**visodom**: Visual Odometry

**XML**: Extensible Markup Language

# 7. References

**[ 1 ] Alexander, Doug and Deen, Robert, Mars Science Laboratory Software Interface Specification, "Camera & LIBS Experiment Data Record (EDR) and Reduced Data Record (RDR) Data Products", JPL D-38107, Mar 4, 2014 (or as updated). This document is included in the PLACES PDS delivery in the DOCUMENTS directory. [http://pds-imaging.jpl.nasa.gov/data/msl/MSLNAV_0XXX/DOCUMENT/MSL_CAMERA_SIS.PDF](http://pds-imaging.jpl.nasa.gov/data/msl/MSLNAV_0XXX/DOCUMENT/MSL_CAMERA_SIS.PDF) .**

**[ 2 ] Deen, Robert, Mars Science Laboratory, "PLACES User Guide", MSL-586-3653, JPL D-71121, Aug 19, 2011 (or as updated).**

**[ 3 ] Collins, Curt, Mars Science Laboratory, Pointing, Positioning, Phasing, & Coordinate Systems (PPPCS), Volume 6, "SA Pointing and Positioning Budgets", MSL-476-1303, JPL D-34648, Dec 1, 2008 (or as updated).**

**[ 4 ] Peters, Steve, Mars Science Laboratory, Pointing, Positioning, Phasing, & Coordinate Systems (PPPCS), Volume 9, "Surface Remote Sensing and Navigation Phasing", MSL-476-1306, JPL D-34651, May 13, 2016 (or as updated). This document is included in the PLACES PDS delivery in the DOCUMENTS directory.**

**[ 5 ] Smith, D.E.; Zuber, M.T.; Frey, H.V.; Garvin, J.B.; Head, J.W.; Muhleman, D.O.; Pettengill, G.H.; Phillips, R.J.; Solomon, S.C.; Zwally, H.J.; Banerdt, W.B.; Duxbury, T.C.; Golombek, M.P.; Lemoine, F.G.; Neumann, G.A.; Rowlands,D.D.; Aharonson, Oded; Ford, P.G.; Ivanov, A.B.; Johnson, C.L.; McGovern, P.J.; Abshire, J.B.; Afzal, R.S.; and Sun, Xiaoli, "2001, Mars Orbiter Laser Altimeter—Experiment summary after the first year of global mapping of Mars" *Journal of Geophysical Research*, v. 106, no. E10, p. 23,689–23,722.**

**[ 6 ] Planetary Data System Standards Reference, February 27, 2009, Version 3.8, JPL D-7669, Part 2.**

# 8. Change Log

This is not intended to be an exhaustive change log, but rather an overview summary of changes in the document.

**Version 1.2 (PDS Release 11)**

- Added DEM cutout image description
- Added `dem_pixel_line` and `dem_pixel_sample` fields to CSV files
- Added `*_demv2` views
- Added conversions tag to XML output
- Added `IMAGE_MAP_PROJECTION` object to map labels
- Added a "`msl_`" prefix to the map filenames to distinguish them better

**Version 1.3 (PDS Release 12)**

- Added reference to the PPPCS vol 9 document [ 4 ] as part of the PLACES delivery.