3D Vision Processing for the MER and MSL Mars Rover Missions

Bob Deen
Jet Propulsion Laboratory
California Institute of Technology
Background

- **Multimission Instrument (Image) Processing Lab (MIPL)**
- **Provides instrument data processing software & services**
  - Multimission and mission-specific
  - Not just images – supports all instrument data
- **Supports most JPL missions**
  - Mars surface, Voyager, Cassini, Magellan, Galileo, Spitzer, Diviner, …
- **Supports both science and operational processing**
- **MIPL created in early 1980’s**
  - Image Processing Lab (IPL) goes back to mid 1960’s
MIPL and Mars missions

• MIPL supports all JPL in-situ Mars missions
  – Viking
    • Completely different software set, not discussed further here
  – Mars Pathfinder
  – Mars Polar Lander
  – Mars Exploration Rovers (MER)
  – Phoenix Lander
  – Mars Science Laboratory (MSL)

• Responsibilities
  – Reconstruction of instrument data from telemetry
  – Systematic creation of derived products (Reduced Data Records – RDR’s) for images
  – Creation of special products for operations, science, and public outreach
  – In the critical path for operations
    • MIPL products required for planning the next Sol’s activities
Mission Planning Cycle

- Assess and Analyze
- Generate Data Products
- Plan Observations and Measurements
- Integrate Activity Plan
- Sequence and Simulate
- Prepare Command Products
- Communicate
- Execute

Opportunity: SOL 272 SKELETON
- 1.3 Ahr
- Predicted Energy Margin
- No deep sleep/2nd Switchback Drive Thursday
- Start: HGA 22720 10:40:00
- End: 11:00:00
- DUR: 0:20:00
- CPU: 1
- Eng Keepout/MTES: 11:02 - 11:15
- Comm Blip: 11:04 - 11:09
- Drive: 11:15 - 11:45
- visodom: 11:45 - 13:15
- Post drive imaging + Tau: 13:15 - 14:00
- Shutdown: 14:00
- NAP: 14:00 - 16:44
- VME on: 0:06
- ODY Mtes & Disable D.S. (e2023): 16:44 - 17:09
- RUHF 42723 (30Mb): 16:52:32 - 17:08:45
- Shutdown: 17:12
- NAP: 17:12 - 3:36
- Wakeup: 0:14
- RUHF 42730 (60Mb): 3:36 - 3:52
- RUHF 42732: 5:22 - 5:35
- Shutdown: 3:52
- Wakeup: 0:06
- SA stomp - heating - cloud observation: 8:00 - 8:45
- Shutdown: 8:45
- Wakeup: 0:06
- AM Science (note: late HGA = longer science): 10:35 - 11:19
- Test (if needed): October 24, 2012
Product Categories

- **Tactical Products**
  - Critical path for daily rover operations
    - Science Planners
      - Define targets and goals, command instrument observations
    - Rover Planners (drivers, arm operators)
      - Generate detailed commands for driving and arm operations
  - Tight timing requirements
    - 1-30 minutes, depending on product

- **Strategic Products**
  - Long-term rover operational planning (days to weeks)
  - Science users (geomorphology, photometry, …)
  - Public release
Product Types

• Over two dozen distinct products per stereo pair
  – Double that if you include L->R and R->L
• Key products:
  – Radiometrically corrected images
  – Geometrically rectified images
  – Disparity maps
  – XYZ images
  – Surface normals
  – Slope maps
  – Reachability/Preload maps (for arm instruments)
  – Roughness maps
• Multiple-image products
  – Terrain meshes
    • Including orbital meshes
  – Mosaics
Software History

- **Scope**: RDR generation programs (except wedge/mesh)
  - Collectively called the Mars program suite
- **Development started for Mars Pathfinder in 1994**
  - MPF-specific, hard-coded parameters, code repetition
- **Software rewritten for Mars Polar Lander**
  - Reusable, mission-independent design
- **Significant upgrades in capability for each new mission**
  - Mission adaptation remains relatively simple
Software Design Overview

• **Suite of 43 application programs**
  – All but 4 are multimission (no mission-specific code!)
    • Exceptions are arm algorithms that are very different for each mission
      – Sometimes use flight software kinematics

• **Built on VICAR image processing system**
  – Core infrastructure: image I/O, parameter processing, O/S isolation
  – Very mature

• **Mission-specific aspects encapsulated into a library**
  – Planetary Image Geometry (PIG)
PIG library

- Object-oriented C++ class library
- Abstracts most functionality needed for in situ missions (rover and lander) into base classes
  - Camera model, pointing, coordinate systems, metadata access, etc.
- Subclasses encapsulate all mission dependencies
  - How to point the MER navcam
  - What a MSL image label looks like
  - How to remove dark current on a PHX image
- Seven missions currently supported
  - MPF, Mars 01 (testbed), FIDO (testbed), Generic, MER, Phoenix, MSL
  - MPL has been obsoleted
  - Software also used for Moonrise and InSight proposal demos and LSOT testbed
- New missions added easily
  - Each amounts to <5% of the code base
  - Adaptation time 1/20 of time to write original library
## RDR generation code

<table>
<thead>
<tr>
<th>Component</th>
<th>Approx. Lines of Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIG Library (Total)</td>
<td>47100</td>
</tr>
<tr>
<td>PIG Multimission Base</td>
<td>22800</td>
</tr>
<tr>
<td>PIG MPL</td>
<td>2400</td>
</tr>
<tr>
<td>PIG M01</td>
<td>1200</td>
</tr>
<tr>
<td>PIG Generic</td>
<td>1200</td>
</tr>
<tr>
<td>PIG FIDO</td>
<td>3000</td>
</tr>
<tr>
<td>PIG MER</td>
<td>4900</td>
</tr>
<tr>
<td>PIG PHX</td>
<td>6100</td>
</tr>
<tr>
<td>PIG MSL</td>
<td>5500</td>
</tr>
<tr>
<td>Applications</td>
<td>99800</td>
</tr>
</tbody>
</table>
Algorithm Overview - Linearization

- **Linearization converts camera model to linear form**
  - Removes fish-eye from hazcams, radial optical distortion
  - Straight lines in the world are straight on linearized images
  - Epipolar-aligns stereo images
    - In practice, results can be up to 5 lines off (Spirit front hazcams)
- **Linearization Pros**
  - Much simpler and faster to compute
  - Models are easier to use
  - 1-D correlators can be used, at least at reduced resolution
- **Linearization Cons**
  - Introduces interpolation noise into images
    - Therefore slightly less accurate
  - Results are not coregistered with EDR’s
  - Linearization done w.r.t. a specific image; must be re-done for another stereo partner
    - e.g. standard vs. long-baseline stereo
- **For MER, all terrain work is done with linearized images**
  - Non-linearized terrains occasionally made as special products (for science requests)
- **For MSL, baseline is to do both**
  - Non-linearized terrains at low compute priority
Raw and Linearized Image

Opportunity front hazcam, sol 2819. Raw on left, linearized on right
Disparity (Correlation)

- **2-D correlation**
  - Compensates for epipolar alignment errors

- **Standard cross-correlation metric**

- **Uses modification of Gruen algorithm**
  - Affine transform + xy terms to map template
  - Amoeba simplex minimization algorithm to determine parameters

- **Consistency check of L->R and R->L correlations**
  - New for MSL, being back-ported to MER

- **Requires starting point for each pixel**
  - Can be reduced resolution; pyramids up to full res
  - 1-D flight correlator at reduced resolution used most commonly
  - Can also use assumed surface, or reversed or unlinearized disparity

- **More sophisticated algorithms possible (SIFT etc) but this works well**
  - Much computer vision research assumes a man-made world
    - Assumptions such as linear walls or sharp corners do not apply

- **Stereo only**
  - Multi-view systems get incredible results but we’re data rate limited
    - Only on rare occasions do we get enough data to consider this

- **Require calibrated cameras**
  - Unconstrained techniques similarly require too much data
XYZ Generation

- **XYZ’s generated by simple geometric triangulation from disparity**
  - Project rays using camera models, find intersection (closest approach)
  - Chooses a point midway between the view rays at closest approach
    - Thus, point not exactly on either view ray
- **Results refined using a series of filters to remove bad points**
  - Missing correlation
  - Excessive raw or average line disparity
  - Not computable, diverging rays
  - Exceeding Z limits
  - Excessive miss distance or miss distance per range
  - Exceeding range limit (based on baseline)
  - Exceeding spike value (too far from neighbors in XYZ space)
  - Remove outliers (isolated points with not enough valid neighbors)
Left: Opportunity navcam, sol 2820; XYZ shows lines of constant X (red) and Y (green) at 1m spacing, with constant Z (blue) at 0.1m.
Right: Front hazcam, sol 2819; range has 1m spacing
Algorithm Overview - others

- **Surface Normals**
  - Fits plane to neighboring pixels, with consistency checks
  - Computed on arm (instrument)-sized and rover-sized patches

- **Slope and Slope-Related Maps**
  - Computed from rover-sized surface normal
  - Slope, slope heading, northerly tilt, solar energy, etc.

- **Arm Reachability Maps**
  - Determines which pixels can be reached by each arm instrument
  - Uses flight software (FSW) arm kinematics and collision models
    - Same algorithms as flight software uses for safety checks
  - Based on XYZ and surface normal
  - Also preload maps, surface roughness
  - Contributed by arm FSW folks
Slope and Reachability Image

Left: Slope from navcam, sol 2965. Colors indicate slope; 0-10 degrees is blue->red.
Right: Arm reachability from front hazcam, sol 2965. Colors indicate different instruments or arm configurations.
Algorithm Overview - Range Error Maps

- New product, still under development
- Per-pixel error estimate
  - Both cross/down range, and axis-aligned
- Given disparity error, project perturbed rays to determine error volume
- Calculating disparity error is currently being worked on
  - Correlation coefficient
  - Compression level
  - Scene activity
  - ...?
- Eventually include terms for camera model error
Algorithm Overview – Meshes

• Converts XYZ point cloud to an octree representation
  – Facilitates merging of multiple XYZ images to unified mesh

• Creates polygonal (triangle) representation of surface
  – Extracts connectivity from XYZ image
  – Uses octree to achieve tiling and multiple resolutions

• Texture Mapping
  – Uses imagery as mesh “skin”
  – Camera model is used to transform 3D mesh vertex -> 2D image coords -> 2D UV texture coords

• Height map (DEM) also produced for driving simulations
  – Simple and fast lookup to settle the rover

• Mesh is in Open Inventor (MSL) or SGI Performer (MER) format
  – Tiled, multiple levels of detail, strips of triangles, binary format
  – Not easily usable by other tools
  – Converter to standard OBJ format is being implemented
Mosaics

• Mosaics are probably the most visible products from MIPL
  – Also most time-consuming task
• Made for many reasons
  – Long-term planning
  – Science use
  – Public release
  • Arguably the signature products of the mission for public engagement
Mosaic Fundamentals

- A mosaic is a single larger image that is made by combining many individual smaller frames.
  - The trick is, to transform and match the images so they look like a unified whole.

- **Requirements**
  - Images have calibrated camera model
    - Transforms XYZ world coordinates to line, sample image coordinates
  - Pointing of each image is known
    - Telemetry from camera mount (e.g. pan/tilt unit)
    - Pointing can be adjusted to reduce seams
  - Traceability of each pixel to source image must be maintained
    - Maintains scientific integrity – quantitative measurements are possible
    - No unconstrained warping
    - No seam blending
MIPL-Supported Projections

- **Cylindrical**
  - Rows are constant lines of elevation, columns constant azimuth
  - Single point of view, generally the center of the ring described by the cameras
  - Standard mosaic for non-stereo in-situ views
- **Polar**
  - Elevation is distance from center (nadir); azimuth goes around the circle
  - Useful for nadir-to-horizon context.
- **Vertical**
  - Rows are lines of constant X, columns are constant Y
  - Overhead view
  - Suffers from severe layover effects when scene doesn’t match surface model
- **Perspective**
  - Models a pinhole camera at a certain point of view
  - Most natural view for small mosaics
  - Can be stereo with appropriate POV
- **Cylindrical-Perspective Hybrid**
  - Each column has its own camera model from its own POV
  - Suitable for stereo panoramas
- **Orthorectified**
  - Uses XYZ data to create “true” overhead view
  - Prototype software
Cylindrical Projection

Opportunity Sol 2820 (Greeley Haven) Navcam, 180 degrees azimuth each
Opportunity Sol 2820 (Greeley Haven) Navcam, North is up
Pancam Color Mosaic

Small portion of Lookout panorama, from Spirit pancam on Sols 410-413
Dealing with Parallax in Mosaics

• Stereo camera consists of camera on either side of a mast head
  – As head is moved in azimuth, cameras describe a circle
  – Each frame is taken from a different position in space, creating parallax
• Fundamental challenge of mosaics is to transform the images so they share a common point of view – that of the output projection
• This could be done perfectly if the camera pivoted about its entrance pupil
  – Images naturally share the same point of view, so no transform is needed and there is no parallax
  – Impractical for stereo-vision cameras
    • A “flagpole” mounting could pivot one camera about its entrance pupil, but stereo partner would have twice as much parallax
• Must project the images to accomplish this transform
Projecting Images

• Transforming the point of view means projecting the image back into 3-D space and then looking at the result from a different POV.
  • If 3-D shape of the scene is known, the projection can be done exactly
    – Basis of orthorectified projections
    – Objects are not distorted, but holes or gores appear in the image due to occlusions
    – Requires stereo analysis of terrain, but stereo not always available, and does not necessarily cover the entire mosaic
  • If 3-D shape is not known, an assumed shape – a surface model – must be used
    – Project image to surface model, view from another point of view
    – Works well if surface model closely matches actual surface
      • Deviations from model cause distortions due to parallax
      • Parallax distortions are based on deviations from surface model, which are usually much less than parallax in the raw images.
Parallax Example

- Surface model can be set to the ground, or the deck
  - One will have seams, the other won’t
  - Can’t eliminate seams in both at once due to parallax
  - Example: Spirit pancams, McMurdo site. Difference is surface model
Stereo Mosaics

• In order to view a mosaic in stereo, separation must be maintained between the left and right eye views
  – Mosaics must be computed from two different points of view

• **Perspective projection**
  – View as from a single camera
  – Put the camera in two suitable places, and the result can be stereo
  – Only works for a limited fields of view (not panoramas)
Panoramic stereo

- **Cylindrical projection cannot be used for stereo panoramas!**
  - Cylindrical projections stem from a single point of view. Move it over for stereo, and it works ahead and behind but you lose stereo separation to the sides.
  - Simply projecting left and right eye views to the same stereo projection does not give proper depth
    - Result is visually a “wall” with bumps on the wall due to deviations from the surface model. Looks very unnatural.

- **Cylindrical-perspective hybrid projection**
  - Each column of output mosaic is a perspective projection from a different point of view.
  - Point of view describes a circle in space as azimuth changes. This maintains stereo separation between the eyes.
  - Stereo looks natural – flat plane extending to the horizon, with height variations on it.
    - Perfect viewing requires tuning disparity at horizon to viewer’s interocular distance (so it looks to be at infinity)
Two cylindrical projections. Foreground appears farther away than horizon over much of the image.

Cylindrical-perspective hybrid. Note how stereo depth matches depth cues and expectations.
Mosaic Correction

- **Geometric seams (discontinuities) are common between images**
  - Parallax
  - Deviations from surface model
  - Imprecise knowledge of camera pointing
- **Seams minimized via a tiepoint-based bundle adjustment process**
  - Gather tiepoints at overlaps between images
    - Manual or automated process
  - BA adjusts parameters to minimize tiepoint error
    - Pointing of individual images
    - Tilt and location of surface model
    - Localization (pose) of rover if it moved during panorama
  - Rigid-body nature of pointing parameters means (adjusted) camera models still apply, so pixels are traceable to their origin.
- **Brightness correction done similarly**
  - Gather brightness/contrast statistics in overlap areas
  - BA adjusts brightness/contrast of each image
  - Loses radiometric calibration, but factors retained so we know what was done to each image
Conclusion

• The Mars program suite has proven successful
  – Integral part of daily ops cycles for MPF, MER, PHX, MSL
  – Algorithms are robust and reliable
• Large software base made manageable by reusable object-oriented design
  – New missions can be accommodated easily
• Work continues to improve it
  – Error metrics
  – Orbital meshes
  – Algorithm improvements
• Questions?
  – Bob.Deen@jpl.nasa.gov