
Level0: DATA IMPORT

make batch list of root filename for batch processing
rm -f bat.lis
ls EDR/*IMG |cut -d"/" -f2 |cut -d"." -f1 > bat.lis

Create ISIS cubes and initialize with SPICE
mkdir -p Lev0/

mdis2isis from=EDR/\$1.IMG to=Lev0/\$1.lev0.cub -batch=bat.lis

output: lev0 cube file

spiceinit from=Lev0/\$1.lev0.cub -batch=bat.lis

*output: Lev0/*cub*

#**Level1 – radiometric calibration**

Create calibrated cubes
mkdir -p Lev1/

mdiscal from=Lev0/\$1.lev0.cub to=Lev1/\$1.lev1.cub -batch=bat.lis

*output: Lev1/*cub*

#**Know your data**

Attach camera statistics and polygon info to image label for qmos;
 # also need footprint polygons for findimageoverlaps

camstats from=Lev1/\$1.lev1.cub attach=true linc=10 sinc=10 -batch=bat.lis

*output: Lev1/*cub*

```
footprintinit from=Lev1/\$1.lev1.cub linc=10 sinc=10 maxemission=89 maxincidence=89 \
      increaseprecision=true -batch=bat.lis

# output: Lev1/*cub

# Create list of level 1 images for further processing
rm  -f  lev1.lis
ls  Lev1/*cub  >  lev1.lis

# Interactive: qmos to view overlaps
# Note: useful to save to a project at this point
```

Establish a ControlNetwork: FINDIMAGEOVERLAPS & AUTOSEED

```
# Compute the individual image overlaps

findimageoverlaps fromlist=lev1.lis overlaplist=overlaps.dat errors=ovl.err detailed=true

# output binary overlap file: overlaps.dat
# error file:  (ovl.err)  if any encountered

#Tip:  The 'overlapstats' application will read the binary overlap file created by findimageoverlaps and report the statistics of the computed
polygons

# Create point locations in the overlaps based on x and y spacing in meters as defined in seed.def

# contents of seed.def:
#Object = AutoSeed
#  Group = PolygonSeederAlgorithm
#    Name = Grid
#    MinimumThickness = 0.0
#    MinimumArea = 0.0
#    XSpacing = 100000
#    YSpacing = 100000
#  EndGroup
#EndObject

autoseed fromlist=lev1.lis deffile=seed.def overlaplist=overlaps.dat onet=auto.net \
```

errors=auto.err pointid="mess_??" networkid=MESS description="Messenger Test"

 # output binary network: auto.net
 # error file: auto.err (even if none encountered)

 # Interactive: ***qmos****:open* **lev1.lis** *to view image overlaps (or open saved project)*
 # open auto.net *to overlay and view tie point positions and density*
 # Notice clustering of points in bottom center - zoom in to see the polygons better

 # NOTE: cnetedit could be run at this point to remove ignored measures and points, but not necessary

---

#Establish the criteria for image references in each control point : CNETREF

 # Interactive: ***qnet -*** *open lev1.lis and auto.net to view points and measures*
 # Toggle through points and their measures (with geom on) to evaluate the variation in resolution, illumination geometry, and point locations

 # The image reference within a point becomes the 'Pattern Chip' or 'truth' for pattern matching

 # For this exercise, set the image based on resolution - the default will select image with mean resolution as reference
 # Include validation of measures based on the DN value (avoid shadows) and how close to the edge of the image that the measure is

 # contents of validmsr.def:
#Group = ValidMeasure
MinDN = 0.002
PixelsFromEdge = 8
#EndGroup

cnetref fromlist=lev1.lis cnet=auto.net onet=auto_ref.net log=auto_ref.log \
deffile=validmsr.def criteria=resolution type=mean

 # output binary network: auto_ref.net, *ascii log file*, auto_ref.log

 # Interactive: ***qmos*** *- open* lev1.lis *to view image overlaps (or open saved project)*
 # open auto_ref.net - *note the* ***red*** *points --> ignored points*

Interactive: ***qnet*** *- open* lev1.lis *and* auto_ref *to see points/measures in more detail*

NOTE: may need/want to consult auto_ref.log for explanation if many points were ignored
Based on results, may need to modify validmsr.def or even rerun autoseed to create a denser network

remove ignored measures/points
cnetedit cnet=auto_ref.net onet=auto_ref_edit.net

output binary network: auto_ref_edit.net

#Pattern Matching (round 1) of image measurements

Subpixel register measurements (pattern matching)

contents of pointreg_P31x31_S101x101.def:

```
#Object = AutoRegistration
#  Group = Algorithm
#    Name          = MaximumCorrelation
#    Tolerance     = 0.7
#  End_Group

#  Group = PatternChip
#    Samples     = 31
#    Lines       = 31
#  End_Group

#  Group = SearchChip
#    Samples = 101
#    Lines   = 101
#  End_Group

#  Group = SurfaceModel
#    DistanceTolerance = 1.5
#    WindowSize       = 7
#  End_Group
#End_Object
#End
```

pointreg fromlist=lev1.lis cnet=auto_ref_edit.net onet=auto_ref_edit_reg.net \

deffile=pointreg_P31x31_S101x101.def flatfile=auto_ref_edit_reg.txt \
points=all measures=all

output: binary network, auto_ref_edit_reg.net; *ascii log file*, auto_ref_edit_reg.txt

 # Notice in pointreg summary that 2 of the 68 points were completely ignored and 50 of the 206
 # measures were ignored/failed. Also notice that FitChipFailures where due to the fit chip
 # tolerance not being met, meaning the Tolerance for the registration was < 0.7.
#....
Group = Points
Total = 68
Ignored = 2
End_Group
Group = Measures
Locked = 0
Registered = 156
NotIntersected = 0
Unregistered = 50
#...

 # Interactive: qmos:open lev1.lis *to view image overlaps (or open saved project)*
 # open auto_ref_edit_reg.net - *note the* ***red*** *points --> ignored points*

 # NOTE: If in research mode, may also open qnet on lev1.lis *and* auto_ref_edit.net *to*
 # modify/fine-tune the autoreg definition file. Use the navigation Filters to find the
 # ignored Points and Measures.

#Pattern Matching (round 2)

 # Subpixel register measurements on Ignored (MeasureType=Candidate) measures/points using a different definition file

 # contents of pointreg_P135x135_S200x200_gradient.def :
Object = AutoRegistration
Group = Algorithm
Name = MaximumCorrelation
Tolerance = 0.2
Gradient = Sobel
EndGroup
Group = PatternChip
Samples = 135

```
#    Lines   = 135
#   EndGroup
#   Group = SearchChip
#    Samples = 200
#    Lines   = 200
#   EndGroup
#  Group = SurfaceModel
#    DistanceTolerance = 1.5
#    WindowSize       = 7
#  End_Group
# EndObject

pointreg fromlist=lev1.lis cnet=auto_ref_edit_reg.net onet=auto_ref_edit_reg2.net \
      deffile=pointreg_P135x135_S200x200_gradient.def  flatfile=auto_ref_edit_reg2.txt \
      points=all measures=candidates

# output: binary network, auto_ref_edit_reg2.net;  ascii log file, auto_ref_edit_reg2.txt

#  There is still 1 point that won't register, but this round picked up an additional 43 measures
#      that were registered.  This was gleaned from the summary.

# Interactive: qmos:open lev1.lis to view image overlaps (or open saved project)
#                  open auto_ref_edit_reg2.net - note the red points --> ignored points

# Interactive: qnet: open lev1.lis and auto_ref_edit_reg2.net and view the results. May decide to
#            manually register the unregistered measures/points at this time or leave as is.

# NOTE: Might need to consult/compare pointreg flafiles to see if measures/points gained in second attempt

# remove ignored measures/points
cnetedit cnet=auto_ref_edit_reg2.net onet=auto_ref_edit_reg2_edit.net

# output: binary network, auto_ref_edit_reg2_edit.net
```

#Quality Control of control network

```
# Validate network by checking connectivity, etc.
```

cnetcheck fromlist=lev1.lis cnet=auto_ref_edit_reg2_edit.net prefix=Chk_ tolerance=0.0

output: ascii files, Chk_Island.1 and Chk_LowCoverage.txt

There are no islands or images with 1 point or less.

Get additional statistics via cnetstats
cnetstats fromlist=lev1.lis cnet=auto_ref_edit_reg2_edit.net \
    create_image_stats=yes image_stats_file=ImgStats_auto_ref_edit_reg2_edit.csv \
    create_point_stats=yes point_stats_file=PtStats_auto_ref_edit_reg2_edit.csv

See how many points on each image, sort by lowest on top
Could also add column 9 to see the convex hull ratio
keep the following commented if attempting to run this as a shell script otherwise the script will stop here until the command is quit

cut -d"," -f1,3 ImgStats_auto_ref_edit_reg2_edit.csv | sort -n -t , -k 2 |more

This network is acceptable for running jigsaw, though it may not be ideal in reality.

#**Jigsaw – BUNDLE ADJUSTMENT Interations**

run jigsaw - no updates!
jigsaw fromlist=lev1.lis cnet=auto_ref_edit_reg2_edit.net onet=JigOut.net \
    update=no radius=no errorpropagation=no outlier_rejection=no sigma0=1.0e-10 maxits=10 \
    camsolve=angles twist=yes spsolve=no \
    camera_angles_sigma=0.025

output: binary network, JigOut.net
ascii output files: bundleout_images.csv, bundleout_points.csv, bundleout.txt

sigma0=0.69 which is good on paper
Sort on residual file to see what is going on there – the 10th column is Vector Magnitude of residual
keep the following commented if attempting to run this as a shell script otherwise the script will stop here until the command is quit

sort -nr --field-separator=, -k 10 residuals.csv |more

maximum if 4.79 pixels, which could be high for this dataset

Interactive: qnet: open lev1.lis *and* JigOut.net. *Use navigation window and filter on Jigsaw Errors*
greater than 3 (or pick a number) and use the information in the Qnet Tool and/or table to
find the measures with high error to verify they are correct or not.

Maximum errors in this network due primarily to due geometric differences and coregistration false
positives. *Could manually fix/ignore bad measures and save to a new filename or remove the*
high residual measures using cnetedit and a valid measure deffile on the jigsaw output n*etwork*.

Typically iterate over jigsaw and adjustments multiple times before updating image, but for the
purposes of this demo, we will use the network as is and update the pointing.

#Jigsaw – Last BUNDLE ADJUSTMENT – Update images

```
# Jigsaw and update images and run error prop
# Need to first stage images for updating if you don't want to re-spice files (good choice)
mkdir -p Updated_Lev1/
rm -f Updated_Lev1/*cub

cp Lev1/*cub Updated_Lev1/
ls Updated_Lev1/*cub > lev1_updated.lis

jigsaw fromlist=lev1_updated.lis cnet=auto_ref_edit_reg2_edit.net onet=JigOut.net \
     update=yes radius=no errorpropagation=no outlier_rejection=no sigma0=1.0e-10 maxits=10 \
     camsolve=angles twist=yes spsolve=no \
     camera_angles_sigma=0.025
# output: binary network, JigOut.net
#      ascii files bundleout_images.csv, bundleout_points.csv, bundleout.txt

# Note on-screen jigsaw results say "Camera pointing updated" and the updated image labels have
#      # Jigged = creation date after the Kernels Group keywords
```

 # Map project the updated images; CAM2MAP

```
mkdir -p Lev2
mkdir -p Lev2/Controlled
rm -f Lev2/Controlled/*cub

# contents of equi_clon180_PosEast_200m.map
# Group = Mapping
#  ProjectionName = Equirectangular
#   TargetName = Mercury
#   EquatorialRadius  = 2440000.0
#   PolarRadius = 2440000.0
#   CenterLatitude  = 0.0
#   CenterLongitude = 180.0
#   LatitudeType = Planetocentric
#   LongitudeDirection = PositiveEast
#   LongitudeDomain = 360
#   PixelResolution = 200
#  End_Group

cam2map from=Updated_Lev1/\$1.lev1.cub to=Lev2/Controlled/\$1.lev2.cub map=equi_clon180_PosEast_200m.map \
      pixres=map -batchlist=bat.lis

# output: Lev2/Controlled/*cub
```

 # Optional: Normalize the photometric properties of the images to achieve uniformity for the map: PHOTOMET

```
mkdir -p Phot
mkdir -p Phot/Controlled
rm -f Phot/Controlled/*cub

# messenger  wavelength specific coefficient values

photomet from=Lev2/Controlled/\$1.lev2.cub to=Phot/Controlled/\$1.lev2.phot.cub \
      maxemission=85.0 maxincidence=89.0 \
      normname=albedo incref=30.0 incmat=0.0 thresh=10e30 albedo=1.0 \
```

phtname=hapkehen theta=17.76662946 wh=0.278080114 hg1=0.227774899 \
hg2=0.714203968 hh=0.075 b0=2.3 zerob0standard=false -batchlist=bat.lis

output: Phot/Controlled/*cub

ls Phot/Controlled/*cub > phot_controlled.lis

 # Create a photometrically normalized mosaic map: AUTOMOS

automos - default priority ontop with tracking tool turned on for image identification in qview

automos fromlist=phot_controlled.lis mosaic=mosaic_controlled.cub track=true matchbandbin=false

#output: image mosaic_controlled.cub

 # Create a 2nd photometrically normalized Averaged mosaic: AUTOMOS priority=Average

#Create a 2nd map where the images have been averaged together
Average mosaics are most useful for showing mis-registrations even after getting good grades (stats) from jigsaw

automos fromlist=phot_controlled.lis priority=average mosaic=mosaic_controlled_average.cub matchbandbin=false

#output: image mosaic_controlled_average.cub

```
# Interactive: qview: mosaic_controlled_average.cub and mosaic_controlled.cub
#            link the mosaics; open tracking tool to note mis-registrations and other notable areas
```

 # Create an Uncontrolled mosaic map (pre-jigsaw, using the original SPICE: CAM2MAP)

```
mkdir -p Lev2
mkdir -p Lev2/Uncontrolled
rm -f Lev2/Uncontrolled/*cub

cam2map from=Lev1/\$1.lev1.cub to=Lev2/Uncontrolled/\$1.lev2.cub map=equi_clon180_PosEast_200m.map \
      pixres=map -batchlist=bat.lis

# output: Lev2/Uncontrolled/*cub

# photomet - this could be skipped if desired
mkdir -p Phot
mkdir -p Phot/Uncontrolled
rm -f Phot/Uncontrolled/*cub

# messenger specific values
photomet from=Lev2/Uncontrolled/\$1.lev2.cub to=Phot/Uncontrolled/\$1.lev2.phot.cub \
      maxemission=85.0 maxincidence=89.0 \
      normname=albedo incref=30.0 incmat=0.0 thresh=10e30 albedo=1.0 \
      phtname=hapkehen theta=17.76662946 wh=0.278080114 hg1=0.227774899 \
      hg2=0.714203968 hh=0.075 b0=2.3 zerob0standard=false -batchlist=bat.lis
# output: Phot/Uncontrolled/*cub

ls Phot/Uncontrolled/*cub > phot_uncontrolled.lis

# automos - default priority ontop with tracking tool turned on for image identification in qview

automos fromlist=phot_uncontrolled.lis mosaic=mosaic_uncontrolled.cub track=true matchbandbin=false
#output: image mosaic_uncontrolled.cub
```

```
# automos - for uncontrol, average mosaics are most useful for showing mis-registrations even
#   after getting good grades from jigsaw

automos fromlist=phot_uncontrolled.lis priority=average mosaic=mosaic_uncontrolled_average.cub matchbandbin=false

#output: image mosaic_uncontrolled_average.cub

# Interactive comparisons between the versions of mosaics created:
#qview mosaic_uncontrolled_average.cub and  mosaic_uncontrolled.cub  mosaic_uncontrolled_average and mosaic_controlled_average
```