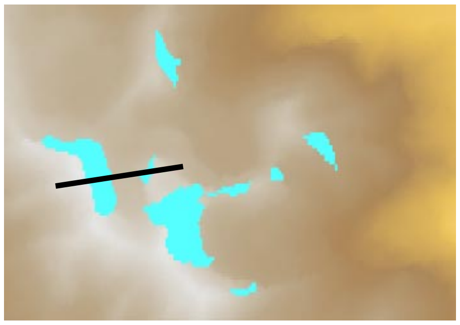
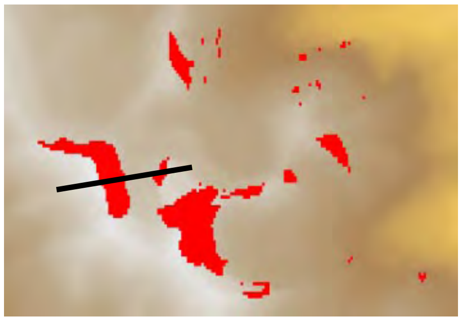
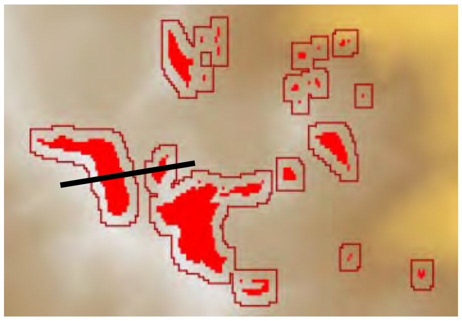
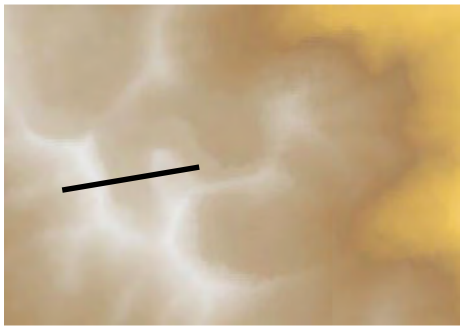
Sample Geospatial Script

Patch Holes in SRTM DEMs


Small portion of a 1-arc second SRTM DEM in a mountainous area. Data holes (null value areas) are shown in cyan. Line of profile shown in black.


Red marks data holes plus SRTM DEM cells that differ in elevation from the reference DEM by more than 75 meters, the difference threshold used in this example for replacing with the DEM value.

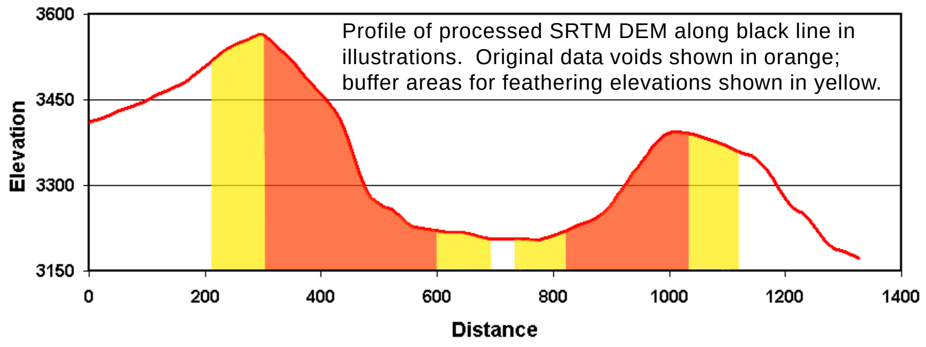

Brown lines outline a 3-cell buffer zone used in this example for feathering the edges of void fills.


Processed SRTM DEM with holes filled and feathered.

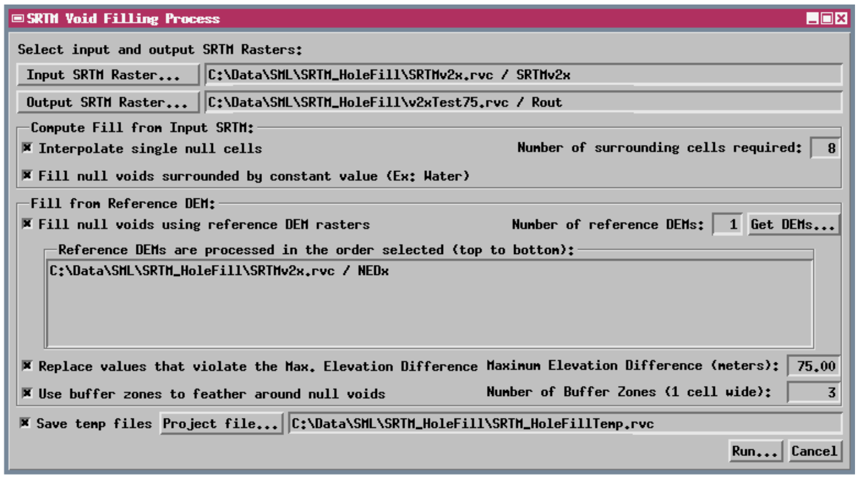
Radar data acquired by the Shuttle Radar Topography Mission (SRTM) has been used by NASA to produce a digital elevation model (DEM) covering 80 percent of Earth's land area. SRTM DEM tiles with 1 arc-second cell size are available for North America, and 3 arc-second DEMs are available for other areas. These SRTM DEMs, however, include data voids (holes) due to localized problems with the radar acquisition process, including layover, shadowing, and lack of signal return from smooth water surfaces.

MicroImages has created a sample geospatial script to fill the no-data cells in SRTM DEMs. The SRTMFILL script (excerpted on the opposite side of this page) operates on the interior null cells in an imported SRTM DEM raster object, filling the holes to provide a continuous elevation surface that can be used for watershed analysis, terrain visualization, or other processes.

Several simple types of holes can be filled using just the SRTM DEM; these holes include single null cells (partly or completely surrounded by real-value cells), which are filled by interpolation, and multicell holes that are surrounded by cells with a constant elevation (such as holes in water surfaces). In addition to these special cases, holes can be filled using one or more reference DEMs, which need not match the SRTM DEM in cell size, geographic extent, or coordinate reference system. Each void cell in the SRTM DEM is replaced with the elevation from the geographically corresponding cell in a reference DEM. You have the option of feathering these hole-fills into their surroundings by specifying a number of 1-cell wide buffer zones around each void. Cell values in each buffer zone are computed by averaging the SRTM and reference cell value using weighting factors that vary across the buffers to provide a smooth blending. You can also choose to set a maximum allowed elevation difference between SRTM and reference DEMs; SRTM values that exceed this difference are initially set to null and are therefore replaced with reference DEM values when the holes are filled.



Profile of processed SRTM DEM along black line in illustrations. Original data voids shown in orange; buffer areas for feathering elevations shown in yellow.


SRTM Void Filling Process

Select input and output SRTM Rasters:

Input SRTM Raster... C:\Data\SML\SRTM_HoleFill\SRTMv2x.rvc / SRTMv2x

Output SRTM Raster... C:\Data\SML\SRTM_HoleFill\v2xTest75.rvc / Rout

Compute Fill from Input SRTM:

☒ Interpolate single null cells — Number of surrounding cells required: 8

☒ Fill null voids surrounded by constant value (Ex: Water)

Fill from Reference DEM:

☒ Fill null voids using reference DEM rasters — Number of reference DEMs: 1 Get DEMs...

Reference DEMs are processed in the order selected (top to bottom):

C:\Data\SML\SRTM_HoleFill\SRTMv2x.rvc / NEDx

☒ Replace values that violate the Max. Elevation Difference — Maximum Elevation Difference (meters): 75.00

☒ Use buffer zones to feather around null voids — Number of Buffer Zones (1 cell wide): 3

☒ Save temp files Project file... C:\Data\SML\SRTM_HoleFill\SRTM_HoleFillTemp.rvc

Run... Cancel

Process dialog window provided by the SRTM Hole Fill script.

Many sample scripts have been prepared to illustrate how you might use the features of the TNT products' scripting language for scripts and queries. These scripts can be downloaded from www.microimages.com/freestuf/scripts.htm.

Script Excerpts for SRTM Hole-Filling Script (SRTMfill.sml)

This procedure traverses through the SRTM raster and replaces any null values with the corresponding cell value from the reference DEM. The SRTM cell will remain null if the current reference DEM does not overlap or if the corresponding reference DEM cell is null.

```
proc FillWithRefRaster() {

    print("Filling Voids with Reference Data\n");

    local class GEOREF outG, refG;
    local class POINT2D point;

    outG = GetLastUsedGeorefObject(outR);
    refG = GetLastUsedGeorefObject(RefR);
     create a transparm if the CRSs are different
    local numeric same = 1;
    if (outG.CoordRefSys.Name != refG.CoordRefSys.Name) {
        same = 0;

        local class TRANSPARM trans;
        trans.InputCoordRefSys = outG.CoordRefSys;
        trans.OutputCoordRefSys = refG.CoordRefSys;
    }

    SetStatusMessage("Filling holes with Reference Data");
     find the overlapping object coordinates for outR
    local class RECT overlap = FindIntersectionOfRasters(outR, RefR, 1);
     traverse through the overlapping area
    local numeric i, j;
    for i = overlap.y2 to overlap.y1-1 {
        SetStatusBar(i, rows);
        for j = overlap.x1 to overlap.x2-1 {
             if the cell is null, find the matching cell in the reference DEM
            if ( IsNull(outR[i,j]) ) {
                point = ObjectToMap(outR, j, i, outG);
                 convert to the correct CRS if they are not the same
                if (!same) {
                    point = trans.ConvertPoint2DFwd(point);
                }

                 convert map coordinates to SRTM raster line/column

                point = MapToObject(refG, point.x, point.y, RefR);
                point.x = round(point.x);
                point.y = round(point.y);

                 check to make sure the point is valid, then replace it

                if ( point.x >= 1 and point.x <= RefR.$Info.NumCols) {
                    if ( (point.y >= 1) && (point.y <= RefR.$Info.NumLins) ) {
                        outR[i,j] = RefR[point.y, point.x] * RefDataScale;
                    }
                }
            }
        }
    }
}
```

This procedure blends the SRTM data with the Reference data that was used to fill the null cells based on the distance from the null voids that the user specifies.

```
proc BlendBufferZone( class RASTER zoneBufferR, numeric zoneNum ) {

    print("   Buffering in Zone " + NumToStr(zoneNum) +
          " of " + NumToStr(cellBufferDist) + "\n");
    local class POINT2D point;
    local class GEOREF outG, refG;

    outG = GetLastUsedGeorefObject(outR);
    refG = GetLastUsedGeorefObject(RefR);
         create a transparm if the CRSs are different
    local numeric same = 1;
    if (outG.CoordRefSys.Name != refG.CoordRefSys.Name) {
        same = 0;

        local class TRANSPARM trans;
        trans.InputCoordRefSys = outG.CoordRefSys;
        trans.OutputCoordRefSys = refG.CoordRefSys;
    }

    SetStatusMessage("Computing new values in Buffer Zone");
     find the overlapping object coordinates for outR
    local class RECT overlap = FindIntersectionOfRasters(outR, RefR, 1);
    local numeric I = 1, J = 1;
     traverse through the overlapping area
    local numeric m, n, buffFactor;
    for m = overlap.y2 to overlap.y1 - 1 {
        SetStatusBar(m, rows);
        for n = overlap.x1 to overlap.x2 - 1 {
             start the blending process if the cell is in the buffer zone
            if ( zoneBufferR[I, J] == 1 ) {
                point = ObjectToMap(outR, n, m, outG);
                 convert point to the correct CRS if not the same
                if (!same) {
                    point = trans.ConvertPoint2DFwd(point);
                }

                 convert map coordinates to SRTM raster line/column
                point = MapToObject(refG, point.x, point.y, RefR);
                point.x = round(point.x);
                point.y = round(point.y);
                 make sure the matching cell in the DEM is valid
                if ( point.x >= 1 and point.x <= RefR.$Info.NumCols) {
                    if ( (point.y >= 1) && (point.y <= RefR.$Info.NumLins) ) {
                         assign the output cell the value of the SRTM plus the
                         linear differnce based on what zone the cells are in
                        buffFactor = (cellBufferDist - zoneNum + 1) * 1.0 / (cellBufferDist
                                * 1.0 + 1);
                        outR[m,n] = outR[m,n] + (buffFactor) * ((RefR[point.y, point.x] *
                                RefDataScale) - outR[m,n]);
                    }
                }
            }
            J++;
        }
        I++;
        J = 1;
    }
}
```