Photogrammetry Notes and Observations

Marvin Smith

October 20, 2015

Contents

Common Variables

- H - Height of the camera above ground, *Flying Height*
- B - Distance between two image, *Air Base*

Common Programs and Usage Notes

This section is dedicated to solving easy problems using common programs.

GDAL Binaries

Many of the included gdal programs can be installed using a package manager.

Ubuntu sudo apt-get install gdal-bin

gdalinfo

Description

gdalinfo is an application bundled with GDAL which provides the user with the ability to extract information about a particular geographic file to the console.

This application works on elevation information, vector files (KML, KMZ), imagery (NITF), and many more.

Information Provided

- Corner Coordinates
- Geographic Projection Used
- Image Raster Datatype
- Date Taken
- more Metadata and image info

Usage

```
./gdalinfo  data/dted/w119/n036.dt2

Driver: DTED/DTED Elevation Raster
Files: data/dted/w119/n036.dt2
Size is 3601, 3601
Coordinate System is:
GEOGCS["WGS 84",
    DATUM["WGS_1984",
        SPHEROID["WGS 84",6378137,298.257223563]],
    PRIMEM["Greenwich",0],
    UNIT["degree",0.0174532925199433],
```

```
    AUTHORITY["EPSG","4326"]]
Origin = (-119.000138888888884,37.000138888888884)
Pixel Size = (0.000277777777778,-0.000277777777778)
Metadata:
  DTED_VerticalAccuracy_UHL=0007
  DTED_VerticalAccuracy_ACC=0007
  DTED_SecurityCode_UHL=U
  DTED_SecurityCode_DSI=U
  DTED_UniqueRef_UHL=G18 063
  DTED_UniqueRef_DSI=G18 063
  DTED_DataEdition=02
  DTED_MatchMergeVersion=A
  DTED_MaintenanceDate=0000
  DTED_MatchMergeDate=0000
  DTED_MaintenanceDescription=0000
  DTED_Producer=USCNIMA
  DTED_VerticalDatum=E96
  DTED_HorizontalDatum=WGS84
  DTED_DigitizingSystem=SRTM
  DTED_CompilationDate=0002
  DTED_HorizontalAccuracy=0013
  DTED_RelHorizontalAccuracy=NA
  DTED_RelVerticalAccuracy=0009
  AREA_OR_POINT=Point
Corner Coordinates:
Upper Left  (-119.0001389,  37.0001389) (119d 0'0.50"W, 37d 0'0.50"N)
Lower Left  (-119.0001389,  35.9998611) (119d 0'0.50"W, 35d59'59.50"N)
Upper Right (-117.9998611,  37.0001389) (117d59'59.50"W, 37d 0'0.50"N)
Lower Right (-117.9998611,  35.9998611) (117d59'59.50"W, 35d59'59.50"N)
Center      (-118.5000000,  36.5000000) (118d30'0.00"W, 36d30'0.00"N)
Band 1 Block=1x3601 Type=Int16, ColorInterp=Undefined
  NoData Value=-32767
  Unit Type: m
```

gdaldem

Description

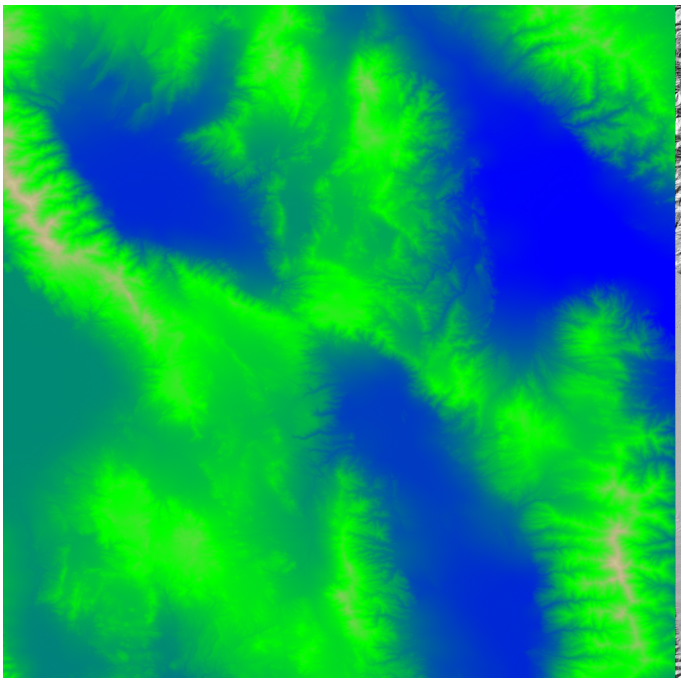
gdaldem is an application bundled with GDAL which provides the user with the ability to construct shaded relief and color maps for digital elevation models.

This application works on elevation information to include DTED and SRTM.

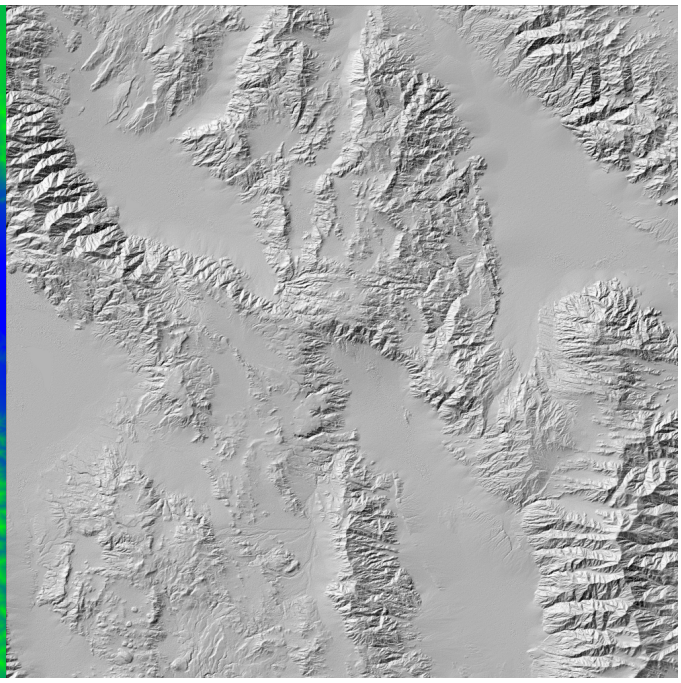
Usage

```
# For color relief maps
gdaldem color-relief data/dted/w118/n36.dt2 color_model.txt output.tif

# For shaded relief maps
gdaldem hillshade  data/dted/w118/n36.dt2 output.tif -s 100000 -z 5

# You can merge the images together with a t=0.7 such that  Out(x,y) = t*Color(x,y) + (1-t)*Hill(x,y)
# to get a great output image
```

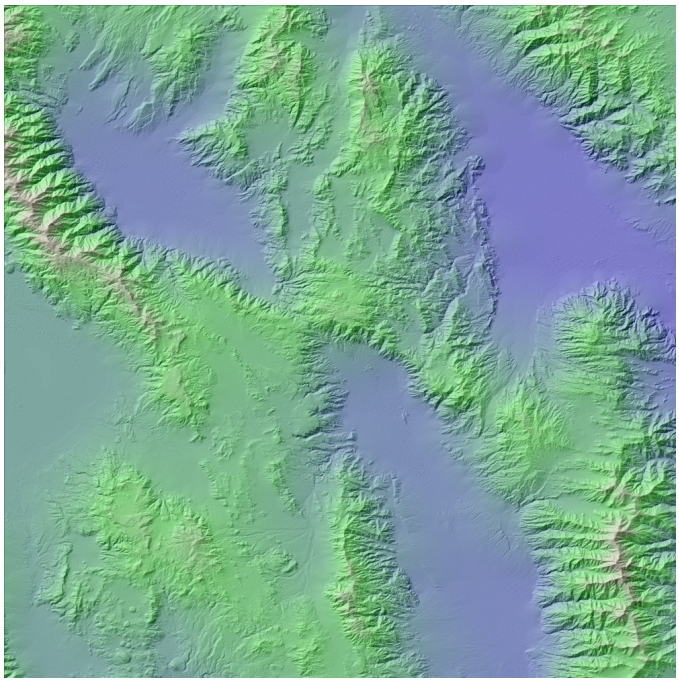
For more information, see manual pages and http://developmentseed.org/blog/2009/jul/30/using-open-source-tools



(a) Color Relief Results



(b) Hillshade Relief Results



(c) Merged Results

Figure 1: Results from typical *gdaldem* usage.

Camera Calibration Notes

Intrinsic Parameters

Common Intrinsic Camera Parameters

- Focal Length (f)
- Radial Distortion

Orthorectification Notes

This section is divided into the following sections···

Camera View Rectification

Camera View Rectification

1. Compute the geographic extents. AKA, find the min and max coordinates and construct an axis-aligned bounding box.
2. Given the bounding box, compute the Ground Sampling Distance. Multiply the GSD with the image to determine the output image size.
3. Iterate over every pixel in the output image.
4. For a pixel in the output image, compute its geographic world coordinate value.
5. Compute the vector which links the pixel's world position with the camera position of the original image.
6. Conduct an intelligent search of the space to look for pixels which occlude the original point.
7. Given the point in world coordinates, compute the pixel in the original image.
8. Set the output image pixel.

Computing the geographic bounding box.

For each corner $P_{\text{pix}}(X, Y)$, compute the world coordinate of the pixel on the surface of the image plane as $P_{\text{img}}(X, Y)$. This will enable you to compute the intersection of the point with the surface plane as P_{world}.

$$P_{\text{img}} = M_w M_c M_p \tag{1}$$

M_p is the transformation which transforms the pixel to the focal plane space. The focal plane space is the 2D image surface originating around the principle point P_o. This is often assumed to be the center of the image. In addition the unit of measure is in meters and the range of the image is the length and width of the CCD surface.

$$M_p = \begin{bmatrix} \frac{X_f}{X_{\text{I}}} & 0 & 0 & -\frac{X_f}{2} \\ 0 & -\frac{Y_f}{Y_{\text{I}}} & 0 & \frac{Y_f}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Where X_f and Y_f are the dimensions of the focal plane and X_I and Y_I are the dimensions of the input image. It is important to note that the Y axis is scaled inversely as image coordinates must be converted from a top-left origin to a bottom-left one.

$$M_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -f & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

M_c converts the pixel space into the camera coordinate system. For convenience, this is merely a 3D coordinate system identical to the focal plane space, where the z axis is the negative focal length. This is important as this will be identical to the world coordinate system except that the camera is rotated into position and translated by the world coordinates of the camera.

$$M_w = \begin{bmatrix} 1 & 0 & 0 & T_{Cx} \\ 0 & 1 & 0 & T_{Cy} \\ 0 & 0 & 1 & T_{Cz} \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{M}_{\mathrm{Quaternion}} \tag{4}$$

Where T_C is the position of the camera in the world coordinate system and $M_{Quaternion}$ is the matrix derived from the quaternion.

This will give you the 3D coordinate of the pixel on the focal plane. You still need to translate this into the actual location on the surface. In order to compute this, you need to compute the intersection between the ray connecting the camera origin and the point on the focal plane with the surface of the Earth.

$$\mu = \frac{N \cdot (P_n - P_c)}{N \cdot (P_p - P_c)} \tag{5}$$

$$P_{\mathrm{gnd}} = P_c - \mu \cdot (P_p - P_c) \tag{6}$$

Where μ is the scale factor of the ray, P_n is the nadir point, P_c is the camera origin, P_p is the principle point, and N is the earth normal pointing $(0, 0, 1)$.

Applying these equations to each of the 4 corners will determine the min and max values for the coordinate system.

Computing the GSD

The ground sampling distance is the distance per pixels, measured in this project as *meters per pixel*. This is computed using the bounding box of the image computed in the previous section. Incorporating the rotation into this would help, however I currently am not sure how to address this.

I currently compute the GSD for the x axis as the average between the span of the top row and bottom row. This span is then divided by the number of pixels. For example, if top row of a 1000 pixel wide image is 5000 meters wide and the bottom row is 1000 meters (the image is rotated), then the GSD is $(5000 + 1000)/1000 = 3$ meters per pixel. The GSD for the Y is the same except column heights. The final GSD is the smaller of the two. A smaller GSD is safer as you can scale the image down and not lose data.

Iterate over image

Once the parameters of the output image are determined, it is necessary to iterate every pixel and find the relationship between its position in the input image. Since we create a bounding box with is axis-aligned, we can easily find the ground position of the pixel in the output image.

$$P_{\mathrm{plane}} = \begin{bmatrix} \frac{\mathrm{bbox.max}.x}{I_{cols}} & 0 & \mathrm{bbox.min}.x \\ 0 & \frac{\mathrm{bbox.max}.y}{I_{rows}} & \mathrm{bbox.min}.y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{7}$$

Where bbox is the geographic bounding box of the image, I is the output image, and X, Y are the image coordinates you are iterating.

From this, compute the plane-line intersection with the image-plane world coordinate and the ground. Use the camera position as the other point.

This equation will yield the ground coordinates of the output image pixel.

Geographic Coordinate to Image Pixel

Now that we know the geographic coordinate of the output pixel, we need to compute the input image coordinate. This will allow us to copy its pixel to the output image. Note that this current technique does zero interpolation, so this should be eventually implemented.

- Compute a line-plane intersection with the line consisting of the camera origin and the ground coordinate. The plane will conist of the camera plane normal and the principle point. This is the point in the world where the camera-ground line intersects the input camera's focal plane.
- Convert the world coordinates into camera coordinates by subtracting out the camera principle point world coordinate. Then rotate to normalize the angle.
- Convert the camera coordinates into 2D image plane coordinates. Multiply the point by the focal plane size, and subtract half. This will bring the point to [0,1] range on the focal plane. Then multiply by the image size to get the real coordinates in pixels.

$$P_{\text{pix}} = \begin{bmatrix} \text{img}_{\text{cols}} & 0 & 0 & 0 \\ 0 & \text{img}_{\text{rows}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & -\text{F}_x \\ 0 & 1 & 0 & \text{F}_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot R_{\text{quat}}^{-1} \cdot \begin{bmatrix} 1 & 0 & 0 & -P_{x_{\text{prin}}} \\ 0 & 1 & 0 & -P_{y_{\text{prin}}} \\ 0 & 0 & 1 & -P_{z_{\text{prin}}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot P_{\text{gnd}} \tag{8}$$

Where F is the dimensions of the focal plane, img are the dimensions of the input image, and P_{prin} is the principle point.

Digital Elevation Model Correction

In order to fix the occlusions caused by the perspective view, it is necessary to search and compute all points on the dem which intersect the line from the camera origin and the world point.

Bibliography

[1] Bon A. DeWitt and Paul R. Wolf. *Elements of Photogrammetry(with Applications in GIS)*. McGraw-Hill Higher Education, 3rd edition, 2000.

Index