Using GDAL/AWS for Historical Aerial Photo Manipulation in a Production Environment

Matthew Wilson
EDR

Data Overview

- 1.4 million historical aerial photos
- 76TB total data
- 1924 to present
- Started collection in middle1990s
- Monthly additions
- Almost 700k images now georeferenced

Previous System

- tabular search results from SQL query (non spatial) in Windows Service application
- image score based on closest to center, but of ballpark coordinates, also uses image quality
- researcher opens images calculated to have best score, crops manually, large images take a long time to load, may need to open multiple images
- research could take 20 minutes for a report with many images!

Data Changes

- New images are all georeferenced.
- Initial search procedure is the same, but with added metadata to prioritize images that are georeferenced.
- Flag fields show where to have the image processing done, either in house or on AWS.
- Run gdalwarp and gdal_translate from existing processes to crop out georeferenced images. Removes need to manually load and crop images.

GDALWARP Command

-
  ```
  “c:\program files (x86)\gdal\gdalwarp.exe" --config
  GDAL_HTTP_UNSAFESSL YES --config
  GDAL_DISABLE_READDIR_ON_OPEN EMPTY_DIR --config
  CPL_VSIL_CURL_USE_HEAD NO  -srcnodata None -cutline
  "PG:host=pegasus dbname=topoDB user=notauser
  password=notapassword" -csql "select * from
  aerialmgr.edr_latlonpointdisttoutmgeom__v4(-
  73.92640,40.82960,628.19300,818.51030,2063,2688)" -
  crop_to_cutline  "/vsicurl/https://edroperations-
  prod.s3.amazonaws.com/edr-
  fullfillment/aerial/17747/1945_10_21_08_008_19450101_20000.ti
  f?AWSAccessKeyId=THISISNOTMYACCESSKEY&Expires=1433517417
  &Signature=THISISNOTTHESIGNATURE"
  c:\temp\aerialcropped\20150604\12467731_567356_1945_0.tif"
  ```

GDALWARP Command Parameters

- **Command**
- c:\program files (x86)\gdal\gdalwarp.exe
- This is just the local version of the gdalwarp command. Keep in mind there are up to five of these processes running simultaneously on the servers running this search, so you will see the gdalwarp and subsequent gdal_translate commands running multiple times.
- **Options**
- --config GDAL_HTTP_UNSAFESSL YES --config GDAL_DISABLE_READDIR_ON_OPEN EMPTY_DIR --config CPL_VSIL_CURL_USE_HEAD NO -srcnodata None
- Have to have an image with an embedded header, no world files will work with this.
- Excellent help from GDALDEV mailing list.
- **Cutline**
- -cutline "PG:host=postGISBox dbname=topoDB user=notauser password=notapassword" -csql "select * from aerialmgr.edr_latlonpointdisttoutmgeom__v4(-73.92640,40.82960,628.19300,818.51030,2063,2688)" -crop_to_cutline
- We are using a shell out to a postGIS function to generate the cutline based on the TP location, desired meter dimensions as well as pixel dimensions. This function takes care of the correction based on pixel dimensions.
- **Source**
- "/vsicurl/https://notmybucketname.s3.amazonaws.com/edr-fullfillment/aerial/17747/1945_10_21_08_008_19450101_20000.tif?AWSAccessKeyId=THISISNOTMYACCESSKEY&Expires=1433517417&Signature=THISISNOTTHESIGNATURE"
- Using vsicurl and the presigned URL, we are able to access the geotiff over https.
- **Destination**
- c:\temp\aerialcropped\20150604\12467731_567356_1945_0.tif
- This is just the locally cropped temporary tiff. We then call gdal_translate to convert it to a jpg for use in our reports.

End Results

- Faster report generation, manual process now becomes a QA process
- Happier customers, fewer misplotted sites
- Better, reuseable data