Optimizing cloud based image storage, dissemination and processing through use of MRF and LERC

Becker, Peter [a]; Plesea, Lucian [b]; Maurer, Thomas [c]

Esri, 380 New York Street, Redlands, CA 92373
[a] PBecker@esri.com, [b] LPlesea@esri.com, [c] TMaurer@esri.com

Commission IV, WG IV/5

KEY WORDS: Raster Format, Image Format, MRF, Compression, Controlled Lossy Compression,

ABSTRACT:

The volume and numbers of geospatial images being collected continue to increase exponentially with the ever increasing number of airborne and satellite imaging platforms, and the increasing rate of data collection. As a result, the cost of fast storage required to provide access to the imagery is a major cost factor in enterprise image management solutions to handle, process and disseminate the imagery and information extracted from the imagery. Cloud based object storage offers to provide significantly lower cost and elastic storage for this imagery, but also adds some disadvantages in terms of greater latency for data access and lack of traditional file access. Although traditional file formats geoTIF, JPEG2000 and NITF can be downloaded from such object storage, their structure and available compression are not optimum and access performance is curtailed.

This paper provides details on a solution by utilizing a new open image formats for storage and access to geospatial imagery optimized for cloud storage and processing. MRF (Meta Raster Format) is optimized for large collections of scenes such as those acquired from optical sensors. The format enables optimized data access from cloud storage, along with the use of new compression options which cannot easily be added to existing formats. The paper also provides an overview of LERC a new image compression that can be used with MRF that provides very good lossless and controlled lossy compression.

1. MANUSCRIPT

1.1 Introduction

The exponential increase in the collection of remote sensing data has created challenges for how to manage, process and serve this data. One significant consideration is how to store the imagery such that it can be quickly accessed and different processing be applied. In traditional image processing workflows the data was download and processed on workstation and the data archived away. Such workflows do not scale and with the advent of cloud computing here is increasing requirement for multiple users and organizations to access the same data. Cloud computing has led to the concept of storing the imagery once and bringing the processing to the storage. Multiple organizations and users can perform processing on the data without needing do download it. Cloud based storage offers to provide significantly lower storage costs and higher durability then is possible with traditional enterprise storage. Its elasticity also enables large numbers of computers to access the same storage.

The way in which imagery data is being accessed has also changing. Instead of uses download original imagery they are accessing the imagery via web services that return only the required data products. The user access mode is changing to use APIs such as ArcGIS REST API, OGC WMS, WCS. The servers which provide this access must though still access the imagery and if stored on cloud storage need to use APIs such as the Amazon S3 REST API.

Another access mode is to pre-process the data into a defined tiling scheme and enable client applications to request and download specified tiles. Only the required tiles are transmitted to the client, which then appropriately display the results. Typically tiled access is used for returning compressed JPEG or PNG tiles for display as background base maps. As the processing capabilities of web clients increases, so does the need to transmit data with greater radiometric resolution and spectral capabilities, which is limited with these compressions.

1.2 Cloud Considerations

Traditional image formats were designed long before cloud computing was conceived. The original objective for the formats was primarily to handle the traditional desktop type access. Moving data and image processing to the cloud presents both an opportunity and a challenge with respect to the storage format. Using transitional image files in cloud storage does not enable the full advantage of the cloud to be realized due to the inherent limitations of the traditional file formats and compression.

There are a number of design criteria that become relevant for raster formats in the cloud:
- The format in which the data is stored needs to be interoperable with multiple applications.
- The data does not necessarily need to remain in its original format, but the structure of the files and metadata do need to remain, else existing workflows that deal with the data break.
- The format should be usable both in cloud storage such as AWS S3 or Azure Block Storage as well as enterprise storage systems.

- The formats need to be able to efficiently handle the very large volumes of numbers of scenes/images/rasters without creating too many files/objects.
- Fast spatial random access in terms of both scale and extent is required.
- Simultaneous direct read of the data is required by many legacy applications
- Support for both rectified and none rectified imagery from satellite, aerial or UAS sensors is required.
- Support is required for modern image sensors and scientific data with high bit depth and a large number of bands.

There are also some valid restrictions that can be placed. For example in most cases scenes from sensors can be assumed to be WORM (Write Once Read Many) in that once written the pixel values are not changed. This has a significant implication in how the data can be cached.

Cloud storage provides additional challenges. Most cloud infrastructure utilizes object storage as low cost storage medium, which is very compelling for the large volumes of imagery data. Object storage is inherently durable and elastic, but has higher latency than traditional file systems and this can influence performance. Access can be optimized by minimizing the number of requests that are made to identify and extract a group of pixels.

1.3 Meta Raster Format

Esri has identified the Meta Raster Format (MRF) designed by NASA JPL as a highly optimal format due to its simple and clean design, cloud optimization, and extensibility.

MRF is a very simple format for tiling imagery. Its original purpose was as a high performance web tile service storage format. MRF is optimized for fast reading and in most implementations splits a raster dataset into 3 separate files:
- Metadata file (.MRF) – XML file containing key properties such as the number of rows & columns, data type, tiling, tile packing, projection and location information. This file is purposely kept small. Additional metadata about the images is stored in separate sidecar files as is required by many satellite vendor products. These include files such with extensions .imd, .dim, .met used for products from WorldView, Pleiades or Landsats scenes.
- Data file – File containing tiles of imagery data at full resolution and optionally with reduced resolution tiles. Tiles may be fully formed raster images such as PNG, JPEG and TIF, or raw data, possibly compressed using Deflate or other compression algorithms. Esri has also added LERC compression as a tile encoding (see below).
- Index (.IDX) – Very simple binary index of tile offsets and sizes within the data file, establishing the geometric organization of the tiles.

Splitting the raster into three files accelerates access to the data tiles, because it enables optimization of the file locations on different classes of storage and it helps with caching. In its simplest implementation, copies of the small MRF and IDX files can be stored on low latency storage, while the data files remains on slower storage. As a result when access random access to a set of tiles is required,, all the required metadata and the index can be quickly read, with only limited data requests to read from the slower storage. Similar when working in cloud environments the metadata and index files can be quickly cached on the machines accessing the data, and then only specific range requests need be made for object storage to access the tiles.

The MRF implementation supports all standard pixel types, bit depths and a large number of individual bands. It also enables an additional third dimension or Z-Slices that can be utilized for some types of multi-dimensional data.

The MRF format is open source and implemented in Geospatial Data Abstraction Library (GDAL see www.gdal.org). Esri has been contributing to its development and has integrated it into ArcGIS 10.4.

MRF provides a way of optimizing access to the millions of scenes from satellite, aerial and UAS sensor. It has a number of advantages over both the more complex traditional file formats, as well as key value map raster implementations such as NoSQL which are more suitable for dynamically changing data sets. MRF does have its limitations. It is not ideal for storing a massive disparate datasets, such as a single raster to define 1m resolution imagery of the entire globe. It is also not optimized for true multi-dimensional datasets or for environments where multiple processors need to write to a single rasters, as may be the case for the output from raster analysis.

1.4 Compression

Compression of the data is important as it reduces volume and cost for storage and transfer of data. The reduction in data transfer volume can speed up access, on the condition that the CPU load required to decompress the imagery is low. One of the issues with some existing compression types is that the CPU load to compress and decompress the image becomes a significant factor in the access speed. In applications where servers are processing massive volumes of data, the decompression costs become significant. Similarly, to enable web clients to directly access the data without plugins, decompression needs to be implemented in the web browsers. Currently JPEG, PNG and GIF are the only generically supported image formats. Other formats need special plug-ins else need to be implementable in JavaScript.

We reviewed what lossy and lossless compression methods are most appropriate for MRF. For imagery of analytical value lossless compression is required. This is true for much of the multispectral imagery from high resolution optical satellites and airborne cameras as well as categorical data such as classification results. A number of lossless compression algorithms exist including lossless JPEG2000, PNG, Packbits, LZW and Deflate. JPEG2000, although typically providing the highest compression, has by far the highest CPU load to compress and decompress. From the other standard compressions, Deflate and LZW provides a good compromise for lossless compression with a relatively low CPU load.

For lossy compression JPEG2000 is a standard that provides good compression, but is very CPU intensive to decompress. JPEG is the most common lossy compression and is very efficient. It is primarily used for 8-bit 3-band imagery and is very fast for typical natural color imagery. It does not provide as high a compression as some wavelet based compression methods, but has the advantage of being directly usable in web applications. A 12bit/channel implementation of JPEG does exist in GDAL as part of the TIF support, and is also supported in MRF. JPEG12 bit is fast to compress and decompress and

has minimal effect on the pixel texture which is important for image correlation use for terrain extraction and segmentation. It is therefore valuable for the compression of panchromatic imagery where the lossy artefacts have minimal effect. One technique for reducing the size of scenes that have a higher resolution pan band, is to compress the pan band using lossy compression while using lossless compression on the multispectral imagery that is used for analysis. On-The-fly pan sharpening can then be used to obtain high resolution multispectral imagery.

Most Lossy compression methods are controlled by a quality parameter that controls the size of the resulting file, but does not control the maximum error of the pixels. 'Controlled Lossy' compression enables a tolerance to be defined that sets the maximum deviation that a compressed pixel may vary from the original value. This enables data to be highly compressed while assuring a suitable precision is maintained. A practical example is the compression of elevation data. Elevation is often stored as floating point, but the source data often contains noise that is beyond the accuracy of the measurements. Such data does not compress well using lossless compression and traditional lossy compression methods result in uncontrolled accuracy degradation.

1.5 LERC – Limited Error Raster Compression

Esri has developed a new compression method called LERC (Limited Error Raster Compression) that was designed to provide such controlled lossy compression, while also being very efficient, such that it utilizes very few CPU cycles both to compress and decompress the data. It does not rely on sequence matching (like LZW, DEFLATE) nor on a space transform (Wavelet, DCT). The algorithm identifies the appropriate scaling to be applied to groups of pixels such that the each group can be quantized and efficiently compressed. The ability to define a tolerance enables it to be used to compress rasters such that the resulting accuracy remains as required, while significantly reducing the storage size.

The compression achieved is highly dependent on the variability of data and the tolerance defined. Typically high resolution elevation data can be compressed between 3-5x higher in comparison to deflate when using a tolerance of 1cm. Compression factors of 5-15x are typically achieved if a tolerance of 50cm is given.

LERC was enhanced to efficiently handle lossless compression of data including both natural color images as well as categorical data. The compression achieved is slightly better than LZW or Deflate. For typical 8bit continuous tone imagery 30% compression is achieved. For higher bit depth satellite imagery such as Landsat 8 lossless compression factors of about 2.5x are achieved, but the factor is dependent on the data content.

The significant advantage of LERC is the compression and decompression speed. For compression the CPU performance is typically about 8x faster than Deflate and 2-3x faster than LZW. For decompression performance is typically 2-3x faster than Deflate or LZW. Both compression and decompression are very significantly faster than JP2000.

Another advantage of LERC is that is has an inherent ability to handle nodata masks. This is important factor in the compression and storage of many rasters especially elevation rasters derived from Lidar and many orthoimages.

LERC also includes check sums that can be used to verify the integrity of the data. This becomes important with the copying or moving of massive data volumes.

LERC can be used to compress imagery stored in file formats, but also for the transmitting blocks of pixels to client applications. This has been implemented in ArcGIS as an optional compression method for the transfer of data values from servers to client applications. The simplicity of LERC has enabled the decoder to be implemented in JavaScript and so also incorporated into web applications that can work directly on the pixel values. This capability is becoming more important as the number of WebGL based applications increase.

LERC is patented, but Esri has specifically released the patent to GIS, terrestrial and extra-terrestrial mapping, and other related earth sciences applications. The source code for LERC has been put into the open source (see https://github.com/Esri/lerc) under an Apache 2 license.

To enable LERC to be used for image and raster storage a container format was required. MRF was found to be an ideal format and Esri has added support for LERC to the MRF format and contributed it to the NASA open source implementation of MRF (see https://github.com/nasa-gibs/mrf)

1.6 Conclusion

MRF provides an optimized format for the storage of imagery in both cloud and enterprise environments. It is advantageous to transform the data to MRF when moving it to cloud or slower access storage environments. MRF has a simple structure that enables high performant implementations. For lossy compression MRF currently utilizes JPEG. For lossless compression None, Deflate, PNG or LERC compression can be currently used. The LERC compression provides further advantages in providing both lossless and controlled lossy compression, while being faster to both compress and decompress.