MRF as a Cloud Optimized Raster Format and LERC Compression

P. Becker

Esri, 380 New York St, Redlands, CA, 92373, USA

17th December 2015

KEY WORDS: Raster Format, Image Format, Compression, Cloud Storage, MRF, LERC, Controlled Lossy Compression

ABSTRACT:

As data management and image analysis move away from a traditional, desktop environment into a cloud-based platform, the file formats which we have relied on until now (such as .tif and .nitf) are no longer optimal because they assumed low latency file access or required a separate server to access the data.

This paper provides details of Meta Raster Format (MRF), a newly evolving image storage format that is optimized for cloud environments, and Limited Error Raster Compression (LERC), an associated compression method that provides faster access to imagery and rasters. Despite the falling cost of storage for large data volumes, there is still significant value in compression. Lossless or controlled lossy compression is necessary so that imagery can be analyzed in an accurate and meaningful way and also to exploit the ever-increasing dynamic range of sensors. Compression decreases the data volumes stored and reduces the data transferred, but there is a tradeoff with the amount of processing power required to decompress. LERC's very efficient algorithm results in very low processing requirements, with very good compression, while maintaining data values within specified tolerance. The decompression can also be implementation in web browsers using JavaScript. The MRF storage format and LERC compression will help resolve some of the challenges of big image data on the internet.

1. MANUSCRIPT

1.1 Accessing imagery

Image processing and analysis has changed significantly over the last 15 years. Traditional desktop image processing packages were designed to process one image at a time. Similarly much image processing was done in a sequential mode one image at a time. The massive increase in computation performance, storage and processing technology has changed image processing and analysis. Cloud infrastructures now enable massive volumes of imagery to be stored and accessed with many processes running in parallel. ArcGIS Image Server is an example of technology that enables large collections of imagery to be quickly accessed with the server applying a wide range of on-the-fly functions to transform the source pixels into valuable information products. These functions apply both geometric and radiometric transformations to the pixels, but require the servers to very quickly access sets of pixels from a large collection of images or rasters. A typical scenario involves a user zooming into an area of interest to visualize a specific band combination from multispectral imagery or the creation of a temporal NDVI (Normalized Difference Vegetation Index) profile for an area of interest.

The dynamic capabilities of Image Server can return requests from a client app that specify the processing to be applied on the original data. A single copy of the source data can thus be used to derive a large range of information products without the need to store the intermediate datasets. In this way the volume of data stored is significantly reduced. Another access mode is to break the data into tiles and enable client applications to request and download specified tiles. Only the required tiles are transmitted to the client, which then performs the required processing and display. Typically tiled access is used for returning JPG 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.

1.2 Cloud Raster Considerations

Traditional image formats were designed long before cloud computing was conceived. The objective for the formats was 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. Once in the cloud the mode of data access relies on HTTP interfaces to extract the required pixels for example with ArcGIS REST API, OGC WMS, WCS or the Amazon S3 REST API.

There are three design criteria 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.
- The original pixel data must be accessible as part of a download process.

The focus changes to ensuring that cloud based processing tools get fast access to the pixels as well as associated metadata. Physically having the data on a personal computer is a lower priority.

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 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.

In addition to the broad design criteria listed above, there are more specific requirements that must also be fulfilled:
- Be accessible from cloud storage such as AWS S3 or Azure Block Storage as well as enterprise storage systems such as NAS and SAN.

- Handle very large volumes of data including large numbers of scenes/images/rasters.
- Enable fast random access in terms of both scale and extent.
- Be able to return many simultaneous requests with direct access and streaming.
- Support both georeferenced of non-georeferenced imagery from satellite, aerial or UAS sensors.
- Support modern image sensors and scientific data with high bit depth and a large number of bands.
- Support different compression methods.
- WORM (Write Once Read Many) access can be assumed as such scenes are typically not modified.

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 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.
- Data file – File containing tiles of imagery data. 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.

The extensions for the files are optional and can be changed if required. Since MRF is a GDAL format, additional metadata not directly handled by MRF but supported by GDAL can be stored in .aux.xml (as defined by GDAL) or other metadata standards defined by source data products. Typically such metadata gets ingested into a database (e.g. a mosaic dataset) and is only accessed from the source during the initial processes that crawl for the metadata. MRF rasters can include internal reduced resolution pyramids with factor 2 or 3, created using nearest or average down sampling.

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 data files remains on slower storage. As a result when access to a tile is required, all the required metadata and the index can be quickly read with only limited data requests to read from the slower storage. In many cases this increase data access performance by 50%. In the GDAL implementation, access to remote files can be achieved using VSICurl.

The MRF GDAL driver is open source and is available on Github. 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 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 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 both the storage costs and transfer volume. 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 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 and 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 especially true for the multispectral imagery from high resolution optical satellites and airborne cameras. A number of lossless compression algorithms exist including lossless JPEG2000, PNG, Packbits, LZW and Deflate. JPEG2000, although providing the highest compression, has by far the highest CPU Load to decompress. From the other standard compressions, Deflate provides a good compromise for lossless compression with a relatively low CPU load.

For lossy compression JPEG2000 is standard that provides good compression, but is very processor 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 supported in ArcGIS. JPEG12 bit is relatively fast to 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 recommendation 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 gain high resolution multispectral imagery as required.

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 most lossy compression methods will 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. Typically high resolution elevation data can be compressed between 3-8x higher in comparison to deflate when using a tolerance of 1cm. Compression factors of 8-20x are typically achieved if a tolerance of 50cm is given.

By setting the tolerance to 0, LERC provides lossless compression. This lossless compression it typically equivalent or better than other lossless compression. LERC also includes an explicit data mask, making it efficient for sparse and projected swath raster data. It also includes check sums that can be used to verify the integrity of the data, which can in some cases be compromised during the copying or moving of massive data volumes. The simplicity of LERC has enabled it to be coded in JavaScript and incorporated into web applications that can work directly on the pixel values. A big advantage of LERC is its performance for both compression and decompression which is significantly faster than other compression algorithms which improves data access and processing.

LERC has been used extensively in ArcGIS for the transmission of elevation data, but has also found to be very effective for the compression of all forms of imagery and has been further enhanced especially to handle 8bit or categorical data.

The source code for LERC has been put into the open source (see https://github.com/Esri/lerc) under an Apache 2 license. LERC is patented, but Esri has released the patent to GIS, terrestrial and extra-terrestrial mapping, and other related earth sciences applications. LERC can be used to compress imagery stored in a file format, but also for the transmitting blocks of pixels to client applications.

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. There are many cases where 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.