i-cubed
content. delivered.

Bryan Carpenter <bcarpenter@i3.com>

# Handling of "axis flip" in gcore/gdaljp2metadata.cpp

**Bryan Carpenter** <bcarpenter@i3.com>
To: Frank Warmerdam <warmerdam@pobox.com>

Fri, May 4, 2012 at 4:36 PM

Hi Frank,

Thanks so much for the information. I think I have figured out what is causing our problem, and I think that the fix I'll show here is likely to be "only good", i.e. not be something that "breaks" some subset of users that are depending on the way it currently is.

I'll discuss the code as it exists here: http://svn.osgeo.org/gdal/tags/1.9.0/gdal/gcore/gdaljp2metadata.cpp in this section beginning at line 762:

```
/* -------------------------------------------------------------------- */
/*      Do we need to flip the axes?                                    */
/* -------------------------------------------------------------------- */
    if( bNeedAxisFlip
        && CSLTestBoolean( CPLGetConfigOption( "GDAL_IGNORE_AXIS_ORIENTATION",
                                               "FALSE" ) ) )
    {
        bNeedAxisFlip = FALSE;
        CPLDebug( "GMLJP2", "Supressed axis flipping based on GDAL_IGNORE_AXIS_ORIENTATION." );
    }

    if( bNeedAxisFlip )
    {
        double dfTemp;

        CPLDebug( "GMLJP2",
                  "Flipping axis orientation in GMLJP2 coverage description." );

        dfTemp = adfGeoTransform[0];
        adfGeoTransform[0] = adfGeoTransform[3];
        adfGeoTransform[3] = dfTemp;

        dfTemp = adfGeoTransform[1];
        adfGeoTransform[1] = adfGeoTransform[4];
        adfGeoTransform[4] = dfTemp;

        dfTemp = adfGeoTransform[2];
        adfGeoTransform[2] = adfGeoTransform[5];
        adfGeoTransform[5] = dfTemp;
    }

    return pszProjection != NULL && bSuccess;
}
```

The swapping of the values at indices [0] and [3] is fine, but I believe that the adjustment of the matrix formed by the values from the two "offsetVector" tags is incorrect here. Here's a representation of the matrix with the values representing the indices in your adfGeoTransform array where the values were placed:

```
1 4
2 5
```

(see this stretch of code at lines 677-682)

```
        adfGeoTransform[0] = poOriginGeometry->getX();
        adfGeoTransform[1] = atof(papszOffset1Tokens[0]);
        adfGeoTransform[2] = atof(papszOffset2Tokens[0]);
        adfGeoTransform[3] = poOriginGeometry->getY();
        adfGeoTransform[4] = atof(papszOffset1Tokens[1]);
        adfGeoTransform[5] = atof(papszOffset2Tokens[1]);
```

The swaps of 1 with 4 and 2 with 5 in the "flip" code above effectively flip the matrix sideways. I believe that the correct thing to do if it is determined that the coordinate order of the data is in fact (y,x) or (lat, long) is to flip both vertically and horizontally, effectively making the swaps be an "X" pattern. To justify this opinion, I'll point out that in normal cases not involving any rotation of the image itself, only the upper left and lower right values will be nonzero. When using (X,Y) or (Long,Lat) ordering, the upper left will be a positive number (for pixels stretching in the positive longitude direction from the upper left corner reference point and the lower right number will be negative, because the pixels of the image stretch downward in the latitude dimension from the upper left reference point.

Related to the fact that an identity matrix (e.g. an Affine transform that has no effect) has a diagonal row of 1's from upper left to lower right, I think it's correct to say that even in the case where the GML is completely specified in (y,x) or (Lat,Long) order, the normal case of the "offsetVector" matrix is to only have nonzero numbers in the upper left and lower right positions. Flipping it sideways makes this cease to be true, and the symptom we see is that the image itself gets flipped about a diagonal line running down and to the right from the reference point. The effect can also be seen in the "Corner Coordinates" section of gdalinfo output. A "wide" image becomes a "tall" image and vice versa.

A completely different approach to explain what should happen to the "offsetVector" matrix is that the first row corresponds to the sentence fragment "**When you move in the image one pixel away from the reference point in <first dimension>...**" and the second row in the matrix represents the sentence fragment "**When you move in the image one pixel away from the reference point in <second dimension>...**"

Correspondingly, the left column in the matrix represents the sentence completion: "**...you are moving this increment in <first dimension>.**" and the right column represents the sentence completion "**...you are moving this increment in <second dimension>.**"

So once again, if you are going to swap the interpretation of dimensions, then you need to flip the matrix vertically followed by flipping it horizontally, and that adds up to individual swaps in an "X" pattern.

Relating to the code above, I think therefore that you want to swap between indices 1 and 5, and then swap between indices 2 and 4.