GFC Canopy Assessment Phase I Methodology Report

Reproducibility of the 2015 Results and a Proposed Method for Future Canopy Analyses

Huidae Cho[1], Ph.D., Jennifer McCullum, Owen Smith

Institute for Environmental and Spatial Analysis, University of North Georgia

December 19, 2019

Abstract

This methodology report studies the 2015 canopy assessment result and investigates its reproducibility. Since the 2009 canopy assessment project aims to use the same data product and method that produced the 2015 deliverable, it is an important step to examine the 2015 reference result and see if it can be reproduced to make sure that the previous method is fully understood. This report discusses findings and solutions, and proposes a new method that is more suitable for future canopy analyses. The new method produces clean and seamless outputs.

Table of Contents

[1] Principal investigator. Email address: hcho@ung.edu

1. Introduction

The objective of the Phase I study is to assess tree canopy in 2009 in the state of Georgia using the same data product and method that were used for conducting the similar 2015 canopy assessment project. A preliminary study was necessary to ensure the reproducibility of the 2015 results. Its goal is to have the 2009 and 2015 datasets use the same model in Feature Analyst to generate the best outcome. Ideally, the original input would be used to reproduce the 2015 results to confirm that the final model is correct. However, the original input data was not available. In substitution, 2015 4-band 1m NAIP imagery was used instead. This dataset was expected to produce identical results considering that the 2015 study also used 2015 4-band 1m NAIP imagery. Unfortunately, the results did not match during the initial testing, which means that the imagery was modified before running the model. After trial and error, it was found that the 2015 study may have reprojected the tiles first, mosaicked them together, and then snapped them to the grid based on a tie point before running the model in Feature Analyst. This information was not helpful for the preliminary study because the tie points cannot be found without the original reprojected images. There was also an overlapping issue due to the fact that USDA added a buffer of 300 meters on all four sides of each NAIP tile. The overlapping and shifting issues are later addressed in this report with a proposed algorithm. Another issue showed that linear artifacts were found along the edges of individual output tiles that are mosaicked into a larger physiographic region. This issue is another reason why the mosaic order is important to mimic similar linear artifacts in the 2009 analysis. Finally, some of the physiographic regions are not aligned and have gaps along their boundaries. With the number of issues presented, a proposed method is recommended that will address most of the issues discussed earlier. However, since it will not follow the same method as the 2015 study, it is strongly suggested that the same method is applied to both the 2009 and the 2015 data to get an outcome with the highest quality. Presented in the following sections are the trial and error methods used in the attempt to reproduce the 2015 results, and the details of the proposed method.

2. Reproducibility of the 2015 Results

2.1. Mosaic Input

For the initial test, the 2015 NAIP tiles are mosaicked to the smallest physiographic region and then ran through the model in Feature Analyst. The results did not match the 2015 study and showed a significant number of cells with different classes as shown in Figure 1.

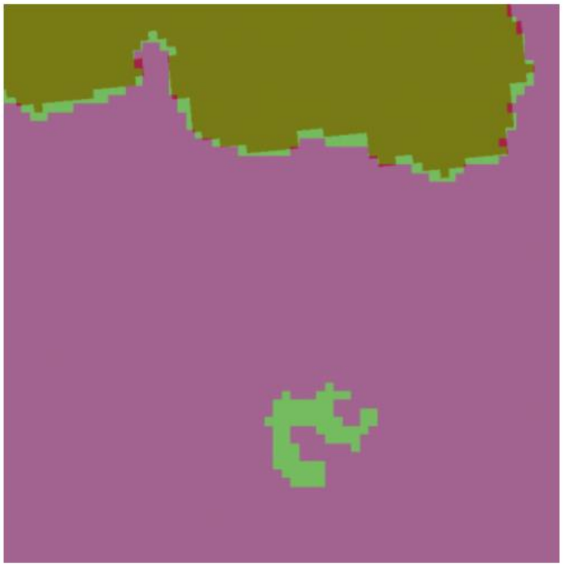


Figure 1. Mismatch between the non-projected output and the 2015 result.

A second attempt was made by reprojecting the mosaicked NAIP imagery before running the model, but there were still a lot of mismatching cells as can be seen in Figure 2.

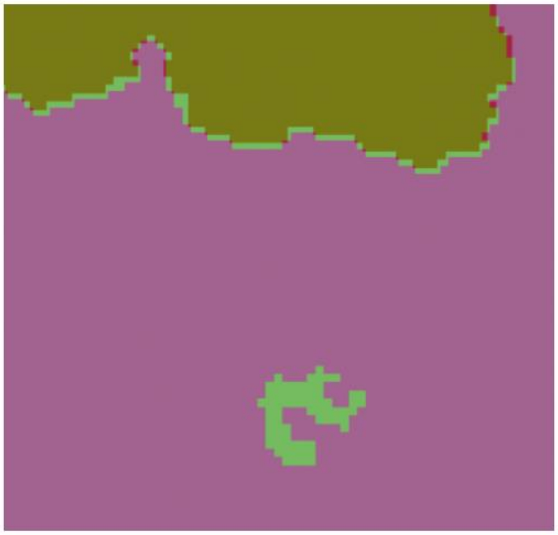


Figure 2. Mismatch between the reprojected output and 2015 results.

Based on this test, it was concluded that the 2015 result may not have been produced using the mosaicked NAIP imagery whether it was reprojected before or after running the model.

2.2. Individual Tile Input

In this test, the model was run on three individual tiles. First, the NAIP tiles were reprojected individually and ran through the model separately without being mosaicked. This resulted in an almost identical shape to the reference output, but there were two problems. First, as shown in Figure 3, there was shifting. Even though the shapes were almost identical, each tile was slightly shifted in a different amount. Addressing this shifting issue could prove challenging when we try to automate the correction process using the Shift tool in ArcGIS. Second, two tiles matched the shape perfectly, but the third tile did not match because the west and east sides of the tile were not the same.

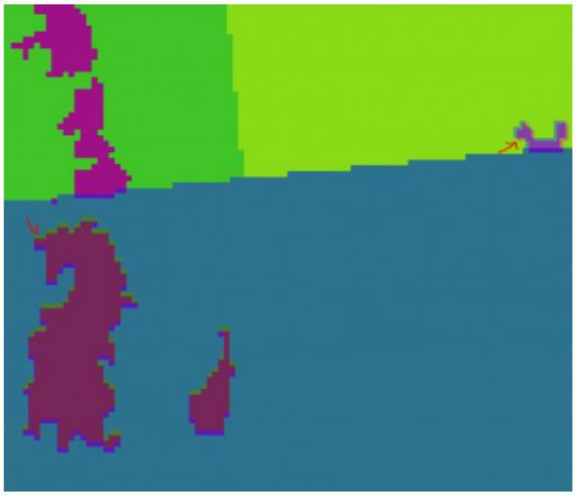


Figure 3. Different shifting in individual tile output.

2.3. Shifting

To better understand the shifting issue, the original NAIP tiles were compared to the reprojected tiles and it was found that they were misaligned. This misalignment led to the conclusion that the shifting issue was caused when the tiles were reprojected as shown in Figure 4. Considering the outcome of this comparison, it was also found that the 2015 study may have reprojected the tiles first, mosaicked them together, and then snapped them to the grid based on a tie point before running the model in Feature Analyst. This information was not helpful for this study because the tie points cannot be found without the original reprojected tiles. Even the 2015 output is clipped to the region boundary and there is no way to figure out where the tie points are. At this point, the only way to figure out the shifting distance is to run the model on individual tiles first and compare their outputs to the reference result.

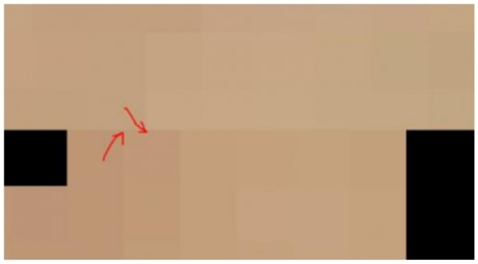


Figure 4. Misalignment between reprojected tiles.

It is not feasible to manually shift 3,913 individual output tiles. To address this issue, an algorithm was used to figure out the shifting distance by running individual tiles and slightly adjusting its cell locations until the adjusted output tile matched the reference result. A trial and error approach was used in Python using the GDAL module independent of ArcGIS. The shift_tiles.py module is available at the gdalutils repository at https://github.com/HuidaeCho/gdalutils.

2.4. Overlapping

USDA added a buffer of 300 meters or 300 pixels (1 meter per pixel) on all four sides of each NAIP tile. This buffer led to overlapping regions because the 2015 product was not clipped around it. The Mosaic to New Raster tool in ArcGIS uses the LAST method by default, which overwrites any cells from previous tiles with those from the last tile. Upon close inspection of the reference output, systematic order of overwriting could not be found and there was little confidence in being able to reproduce the same order of mosaicking without the original reprojected tiles. Figure 5 clearly shows this issue.

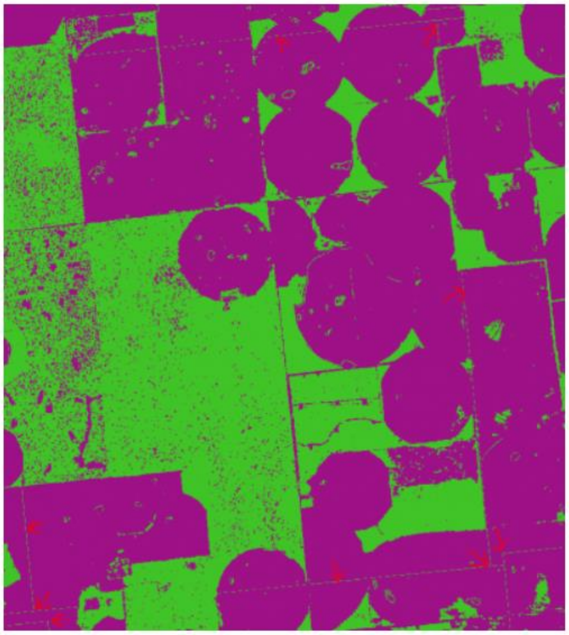


Figure 5. Overlapping mosaicked tiles in the 2015 deliverable.

Tackling the stacking order problem can be difficult if it is done from the "ordering" aspect of mosaicking because 3,913 output files need to be sorted through. A proposed method redefines the problem as a "masking" problem. That is, a mask will be created for each output tile by finding which cells in each output tile were overwritten by neighbor tiles and which cells were not, instead of trying to figure out the same order of mosaicking. The proposed algorithm works as follows:

```
1. For each tile,
   a. Create a dictionary called cells.
   b. Create a matrix called mask of the same dimension as the current tile with value 0.
   c. Read the SE corner of the NW tile that overlaps the current tile into matrix cells[1] of the same dimension as the current tile.
   d. Add 1 to the mask cells that have cell values in the above cells matrix.
   e. Read the S side of the N tile that overlaps the current tile into matrix cells[2] of the same dimension as the current tile.
   f. Add 2 to the mask cells that have cell values in the above cells matrix.
   g. Read the SW corner of the NE tile that overlaps the current tile into matrix cells[4] of the same dimension as the current tile.
   h. Add 4 to the mask cells that have cell values in the above cells matrix.
   i. Read the W side of the E tile that overlaps the current tile into matrix cells[8] of the same dimension as the current tile.
   j. Add 8 to the mask cells that have cell values in the above cells matrix.
   k. Read the NW corner of the SE tile that overlaps the current tile into matrix cells[16] of the same dimension as the current tile.
   l. Add 16 to the mask cells that have cell values in the above cells matrix.
   m. Read the N side of the S tile that overlaps the current tile into matrix cells[32] of the same dimension as the current tile.
   n. Add 32 to the mask cells that have cell values in the above cells matrix.
   o. Read the NE corner of the SW tile that overlaps the current tile into matrix cells[64] of the same dimension as the current tile.
   p. Add 64 to the mask cells that have cell values in the above cells matrix.
   q. Read the E side of the W tile that overlaps the current tile into matrix cells[128] of the same dimension as the current tile.
   r. Add 128 to the mask cells that have cell values in the above cells matrix.
   s. Find unique cell values in mask.
   t. For each unique cell value x,
      I. For each of 1, 2, 4, 8, 16, 32, 64, 128 in y,
         A. If cells[x&y] dominates this unique cell value region,
            i. Set the cells in mask with value x to 256|y.
            ii. Move to the next unique cell value.
```

```
        II. If there were no dominating cells[x&y],
            A. Set the cells in mask with value x to 256.
    u. Set mask to the logical AND of mask and NOT 256.
    v. mask is the final mask with 1, 2, 4, 8, 16, 32, 64, or 128 indicating which neighbor tile dominates
       where.
```

This algorithm assumes that different tiles do not have the same cell values everywhere they overlap. It is a reasonable assumption because neighboring NAIP tiles are likely to have different 4-band values in the same area. The algorithm also assumes that the extent and shape of the same tile from different years stay unchanged because it creates raster masks. The latter assumption can be problematic even if there are only slight changes in size along the edges of NAIP tiles. To address this potential issue, a dictionary was created that will contain information about the non-dominating corners and sides of each tile instead of creating a hard-shaped raster mask. This revised algorithm works as follows:

```
1. Create a dictionary called dominated by
2. For each tile,
    a. Create a dictionary called cells.
    b. Create a matrix called mask of the same dimension as the current tile with value 0.
    c. Read the SE corner of the NW tile that overlaps the current tile into matrix cells[1] of the same
       dimension as the current tile.
    d. Add 1 to the mask cells that have cell values in the above cells matrix.
    e. Read the S side of the N tile that overlaps the current tile into matrix cells[2] of the same dimension
       as the current tile.
    f. Add 2 to the mask cells that have cell values in the above cells matrix.
    g. Read the SW corner of the NE tile that overlaps the current tile into matrix cells[4] of the same
       dimension as the current tile.
    h. Add 4 to the mask cells that have cell values in the above cells matrix.
    i. Read the W side of the E tile that overlaps the current tile into matrix cells[8] of the same dimension
       as the current tile.
    j. Add 8 to the mask cells that have cell values in the above cells matrix.
    k. Read the NW corner of the SE tile that overlaps the current tile into matrix cells[16] of the same
       dimension as the current tile.
    l. Add 16 to the mask cells that have cell values in the above cells matrix.
```

m. Read the N side of the S tile that overlaps the current tile into matrix cells[32] of the same dimension as the current tile.

n. Add 32 to the mask cells that have cell values in the above cells matrix.

o. Read the NE corner of the SW tile that overlaps the current tile into matrix cells[64] of the same dimension as the current tile.

p. Add 64 to the mask cells that have cell values in the above cells matrix.

q. Read the E side of the W tile that overlaps the current tile into matrix cells[128] of the same dimension as the current tile.

r. Add 128 to the mask cells that have cell values in the above cells matrix.

s. Find unique cell values in mask.

t. Sort them by the number of values in ascending order.

u. Create a list called overwritten_by.

v. For each unique cell value x,

```
    I. For each of 1, 2, 4, 8, 16, 32, 64, 128 in y,
        A. If cells[x&y] dominates this unique cell value region,
            i. Append y to overwritten_by.
            ii. Move to the next unique cell value.
    II. Add overwritten_by to dominated_by[tile_id].
```

w. dominated_by contains overlapping information overwritten_by for each tile that includes 1, 2, 4, 8, 16, 32, 64, or 128 indicating which neighbor tile dominates in which order.

2.5. Linear Artifacts

Linear artifacts were found along the edges of individual output tiles that are mosaicked into a larger physiographic region. These linear artifacts have been introduced by Feature Analyst because there is no information beyond the edges of a tile. Since the 2015 output did not remove the extra buffer around each output tile, these artifacts remained in the final product. This issue is another reason why the mosaic order is important to mimic similar linear artifacts in the 2009 analysis.

USDA provides boundary polygons for individual NAIP tiles that do not include the 300-meter buffer area. Since these linear artifacts are only created along the edges of tiles, which are outside the main tile region, they can be clipped off from the main area of each tile before mosaicking. The proposed method implements this clipping step.

2.6. Misaligned Physiographic Regions

Finally, some physiographic regions are not aligned and have gaps along their boundaries as shown in Figure 6. Ideally, there should not be any gaps between output physiographic regions.

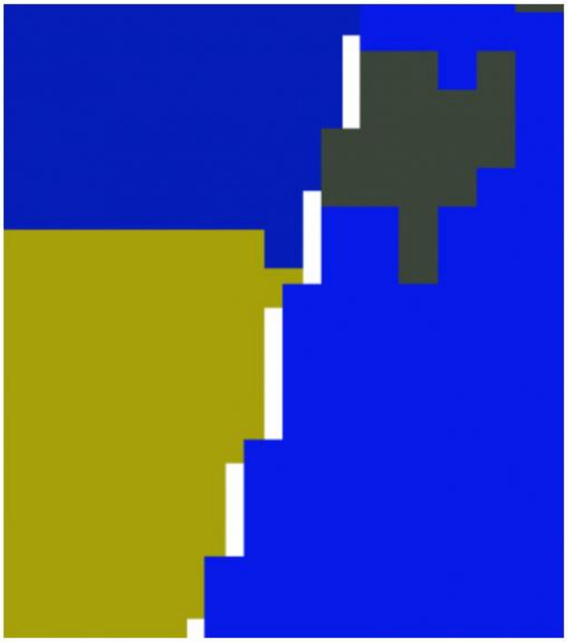


Figure 6. Cells misaligned between physiographic regions and NoData gaps in the 2015 deliverable.

These gaps could be removed if all of the output tiles were snapped to the same grid system. In the proposed method, the very first output tile will be used as a reference raster for snapping the other tiles afterwards.

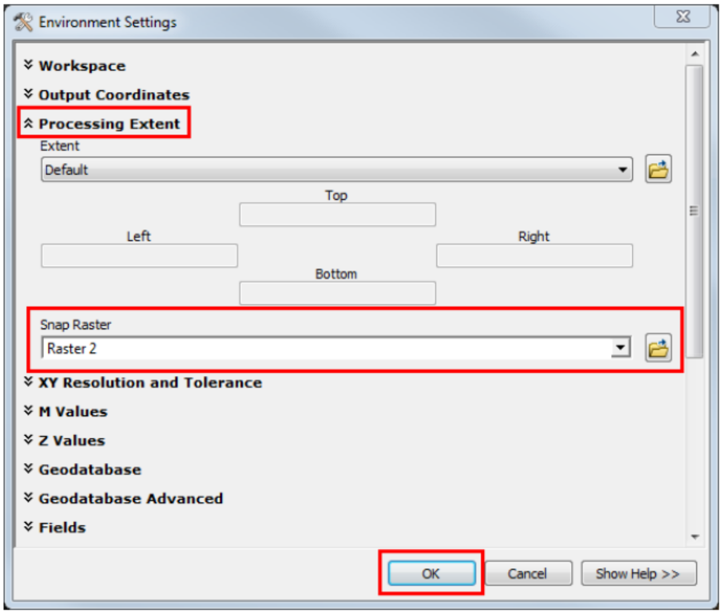
3. Proposed Method for Future Canopy Analyses

The following method is proposed to address most of the issues discussed earlier. However, since it will not follow the same method as the 2015 study, it is strongly suggested that the same proposed method is applied to both the 2009 and the 2015 data to get an outcome with the highest quality.

<u>Step 1. Reproject the individual NAIP tiles</u>

1. Use the Project Raster tool in ArcGIS.
2. Change the projection to the USA Contiguous Albers Equal Area Conic USGS version projection (SR-ORG:6703, SR-ORG:7480, WKID 102039). This projection follows GFC's convention.
3. Since Georgia is split into two UTM—NAIP projection—zones, it would be better to deliver the final product in one common projection.
4. Do not snap the first tile as there will not be any reference raster for snapping in the new projection.

5. For the other tiles, use the very first reprojected tile as a snap raster in the Environment setting shown in Figure 7.



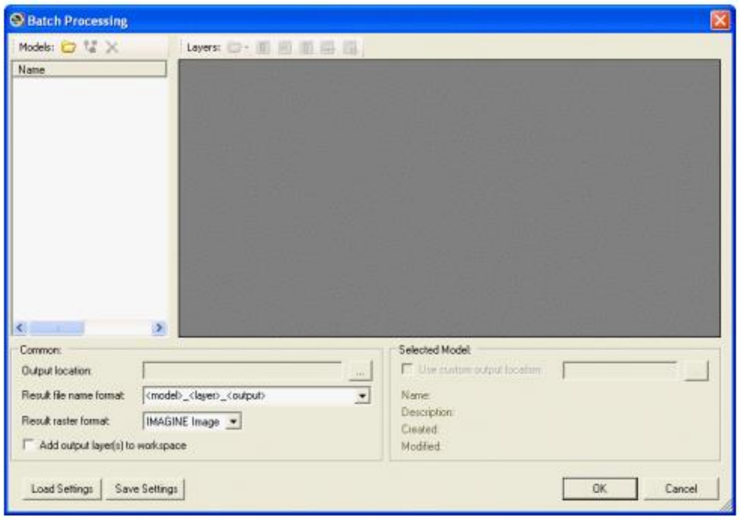
Extent: Default

Snap Raster: Raster 2

Figure 7. ArcGIS Environment Settings.

Step 2. Run the reprojected individual tiles in the Feature Analyst model

1. Open the Feature Analyst toolbar and select Learning ➔ Batch Processing. The dialog box in Figure 8 will display.



Output location:

Result file name format: <model>_<layer>_<output>

Result raster format: IMAGINE Image

□ Add output layer(s) to workspace

□ Use custom output location

Name:

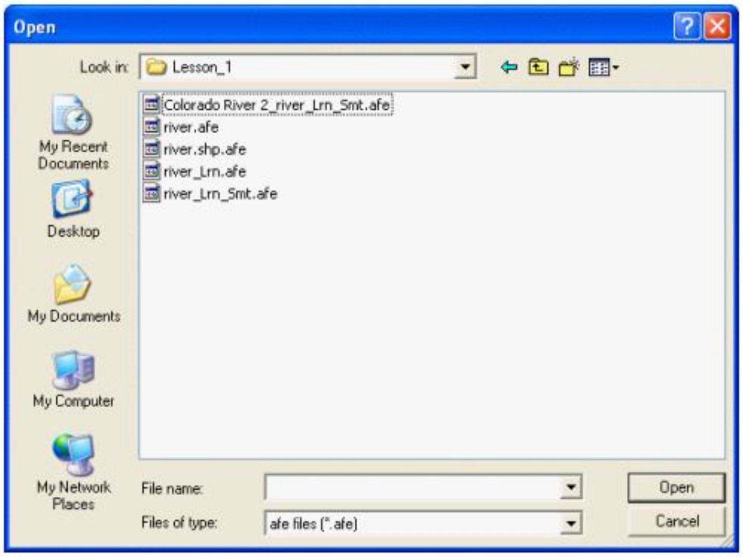
Description:

Created:

Modified:

Figure 8. Batch Processing dialog box.

2. Click on the Add Models button, select the Feature Analyst model, and click open (Figure 9).



Look in: Lesson_1

Colorado River 2_river_Lrn_Smt.afe
river.afe
river.shp.afe
river_Lrn.afe
river_Lrn_Smt.afe

File name:

Files of type: afe files (*.afe)

Figure 9. Select the Feature Analyst model.

3. The model becomes active when the input layer is added to batch processing.
4. Next, right-click the Input raster layer to add the NAIP tile to be processed.
5. Click OK to run the batch processing.

Step 3. Clip the output tiles to the NAIP tile polygons

1. Use the Extract by Mask tool to clip out the buffer area where linear artifacts were generated by Feature Analyst.

Step 4. Mosaic the clipped output tiles

1. Use the Mosaic tool in ArcGIS to mosaic individually clipped output tiles. Since the output tiles are already clipped, there will be no overlap between the mosaicked tiles.

4. Results and Discussion

Figure 10 clearly shows that the final output has significantly been improved over the 2015 result. Both figures show the mosaic result of the same region where four NAIP tiles overlap. The figure on the left produced by the proposed method is seamless while the figure on the right from the 2015 result has linear artifacts introduced by Feature Analyst and the mosaicking process. Even in the proposed method, Feature Analyst still generated linear artifacts along the edges of input tiles, but they were located outside the NAIP polygon of each tile. The clipping step removed those artificial features and mosaicking the clipped output tiles generated a seamless mosaicked output.

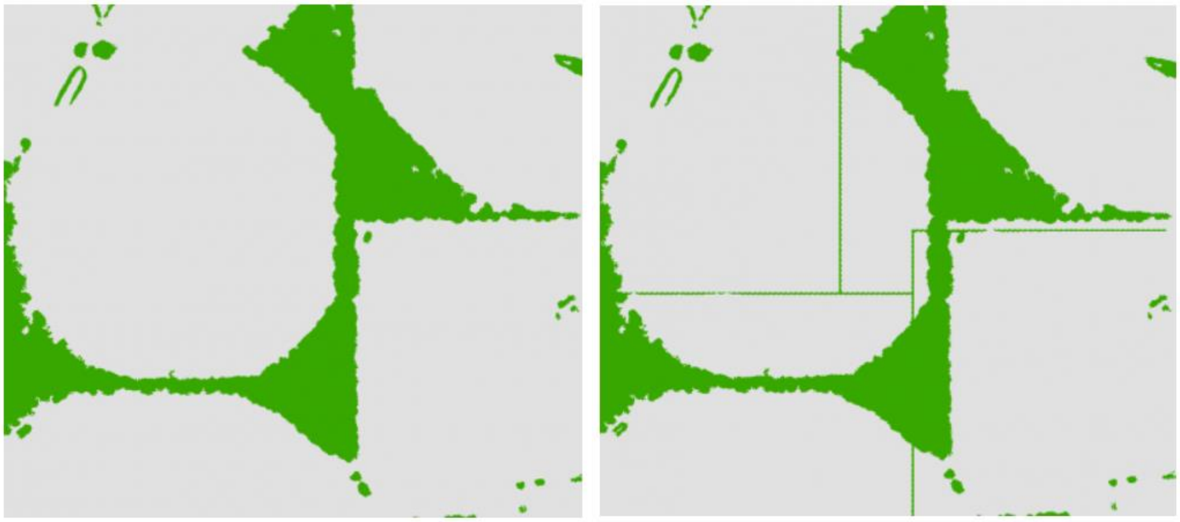


Figure 10. Same area using the new method (left) and the 2015 method (right).

Figure 11 shows a clean and seamless mosaicked output. This test result does not have any of the issues discussed earlier.

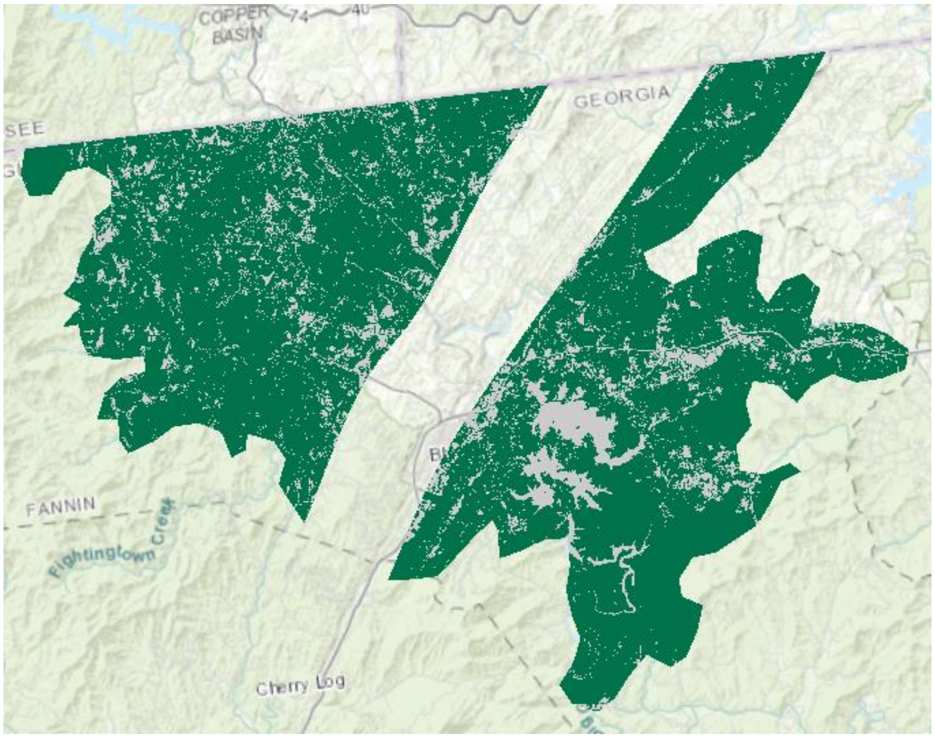


Figure 11. 2009 result for the McCaysville Basin physiographic region.

5. Conclusions

In this preliminary methodology study, the 2015 canopy result was examined and some issues were found. Most of these issues might mainly have been caused by inconsistent mosaic snapping. Most importantly, it was not possible to reproduce the same result using the same input and method because of the mosaicking process with overlapping regions. Since the overlapping regions from neighboring tiles do not necessarily have the same data, it is problematic to mosaic them without any systematic mosaicking order. A new method was proposed for future canopy analyses. The proposed approach properly handles the overlapping regions and mosaic snapping. A pilot test was performed using the new method and it was confirmed that the test output is seamless without the same issues.