Improvements to DEM Merging with `r.mblend`

Luís Moreira de Sousa[1] and João Paulo Leitão[2]
[1]*ISRIC - World Soil Information, Droevendaalsesteeg 3, Building 101, 6708 PB Wageningen, The Netherlands*
[2]*Swiss Federal Institute of Aquatic Science and Technology (EAWAG),Urban Water Management Department (SWW), Überlandstrasse 133, CH-8600, Dübendorf, Switzerland*

Keywords: Digital Elevation Model (DEM), Terrain Analysis, Raster.

Abstract: `r.mblend` is an implementation of the *MBlend* method for merging Digital Elevation Models (DEMs). This method produces smooth transitions between contiguous DEMs of different spatial resolution, for instance, when acquired by different sensors. `r.mblend` is coded on the Python API provided by the Geographic Resources Analysis Support System (GRASS), being fully integrated in that GIS software. It introduces improvements to the original method and provides the user with various parameters to fine tune the merging procedure. This article showcases the main differences between `r.mblend` and two conventional DEM merge methods: *Cover* and *Average*.

1 INTRODUCTION

In the Geographic Information Systems (GIS) domain, the representation of terrain elevation has been predominantly performed using the raster data format, in what are called Digital Elevation Models (DEMs). The discretisation of elevation by a regular grid is rather useful in software development, with a direct correspondence to a two dimensional array. This ease of development has fostered the creation of numerous spatial analysis methods (de Smith et al., 2015), making DEMs ever more convenient.

DEMs have traditionally been acquired by stereoscopic sensors on board of air-borne or space-borne vehicles. For decades DEMs remained an expensive and inaccessible type of data. The emergence of technologies like Light Detection and Ranging (LiDAR) sensors, and small and easy to operate Unmanned Aerial Vehicles (UAVs) have made the acquisition of high resolution DEMs considerably simpler and inexpensive (Küng et al., 2011).

With multiple DEMs obtained by different methods and at different spatial resolutions available, spatial analysts often face today the need to combine or merge various of these data sets. However, there is no obvious method for doing so; a direct merge of overlapping DEMs produces artefacts along borders, leading to inconsistent terrain aspects and slopes (Katzil and Doytsher, 2005; Luedeling et al., 2007). Spatial analysis conducted on such merged DEMs inevitably results in fickle results, be it in view-shed computation, overland water flow, least cost path, etc.

The *MBlend* method (Leitão et al., 2016) proposes to merge two overlapping DEMs by retaining the highest spatial resolution DEM and introducing a smooth transition into the lower resolution DEM. Modifications are applied only to the lower resolution DEM, producing a single DEM that covers the entire study area with the highest possible accuracy, while also ensuring smooth transitions between the original DEMs. `r.mblend` is an implementation of the *MBlend* method, coded in the Python language as an add-on to the Geographic Resources Analysis Support System (GRASS). It introduces an advanced and flexible computation of the transition between DEMs, that the user may tune through various parameters.

This article compares results obtained using `r.mblend` with those of two conventional DEM merging methods on two different test cases. Section 2 describes the methods used, Section 4 presents the test cases used for comparison and Section 6 rounds up the results.

2 RASTER MERGING

2.1 Conventional Methods

Common GIS programmes provide simple functions to merge raster data sets. They usually require the inputs to have the same cell size and be in

Sousa, L. and Leitão, J.
Improvements to DEM Merging with `r.mblend`.
In *Proceedings of the 4th International Conference on Geographical Information Systems Theory, Applications and Management (GISTAM 2018)*, pages 42-49
ISBN: 978-989-758-294-3

the same coordinate system. These simple methods can be classified in two different types: *Cover* and *Average* (Eastman, 2012).

Cover type methods do not operate any adjustments to the input DEMs, they are simply superimposed. The DEM resulting from this method assumes cell values of the first input across its entire extent and values from the second input in areas not covered by the first. The resulting DEM can yield significant elevation discontinuities along the boundary between the input DEMs, resulting in erroneous slope and aspect values (Hickey, 2000).

Average methods assign to the merged DEM the average elevation within areas where the input DEMs overlap. Outside the overlapping area the resulting DEM assumes the value of the existing input; only values within the overlapping area are changed. Some of these methods try to tackle the discontinuities issue using a weighted average, as is the case with IDRISI (Eastman, 2012). A subset of these, usually referred as *Blend* methods, go further, using a averaging weighting function that may be linear, smoothed (e.g. bicubic), or discontinuous; this way more weight can be given to one of the inputs in certain areas, e.g. closer to borders. It must be noted though, that these averaging methods act as low pass filters, therefore reducing the accuracy of the resulting DEM.

2.2 *MBlend*

The *MBlend* method differs from conventional methods in two essential aspects: it is aware of the different spatial resolution of its inputs and modifies areas for which only low resolution data are available (Leitão et al., 2016). This method works by identifying two edges: the border between the low and high resolution inputs (near edge) and the border around the area of the low resolution input not overlapping with the high resolution input (far edge). Points are set along each of these two edges, those on the near edge take the difference between the two inputs at the location; those on the far edge take the value zero (see Figure 1). From these points is computed a transition surface, spatially restricted to the area of the low resolution input, not overlapping with the high resolution input. Finally, the transition surface is subtracted from the low resolution input; the resulting DEM assumes the values of the high resolution input within its extent and outside the values of the low resolution input minus the transition surface.

The *MBlend* method consist in seven essential steps:

1. *Obtain the low resolution extent* - this is the extent of the study area that is only covered by the low resolution DEM. It can be obtained by vectorising the extent of each DEM and then applying an intersection.
2. *Compute differences* - obtained by subtracting the low resolution from the high resolution DEM.
3. *Obtain the* near edge - the differences map is vectorised into points. A buffer around the low resolution extent is then used to select from these difference points those that lay along the border between the two rasters (see Figure 1 (a)).
4. *Obtain the* far edge - the low resolution DEM is vectorised to points and those along the border are selected using an internal buffer to the low resolution extent.
5. *Build interpolation points set* - the value zero is assigned to the points in the *far edge*; it is then merged with the *near edge* into a single data set.
6. *Interpolate smoothing surface* - a new raster surface is created by interpolation using the edges points data set. The resulting surface smoothly transitions from the full difference between the two input DEMs along the *near edge* towards zero along the *far edge* (see Figure 1 (b)).
7. *Apply smoothing* - the smoothing surface is added to the low resolution raster. The result is then patched with the high resolution raster to obtain a single data set covering the entire study area.

3 Implementation

3.1 The GRASS Add-on Development Environment

GRASS is a Geographical Information System (GIS) originally developed by the US Army Corps of Engineers with a focus on spatial data management and analysis (Neteler et al., 2012). It is characterised by a deep dataset management and archiving structure and a vast roll of analysis operations, also known as modules. GRASS manages multiple data-set types: raster, vector, imagery and voxel (3D).

GRASS was originally written in C, with its modern structure now also coded in C++. Since 2012, in the wake of version 6.4.2[1], an API to the GRASS C library was made avaialble for the Python programming language (Sanner et al., 1999). This API greatly simplified the development of new GRASS modules,

[1]https://grass.osgeo.org/announces/announce_grass642.html

also facilitating the integration of popular Python libraries such as NumPy[2] or Pandas[3].

A system to host new modules – called "add-ons" – was also created, whereby third party developers commit their code to the GRASS repository, thus making their module(s) automatically available to all GRASS users. These "add-ons" can be added to every GRASS installation with the module `g.extension`. This module connects automatically to the GRASS repository, downloads and installs the required binaries or code.

Developed within this environment, `r.mblend` is versioned and managed at GitHub[4], and released under the European Union Public Licence[5].

3.2 Improvements Over the Original `MBlend` Method

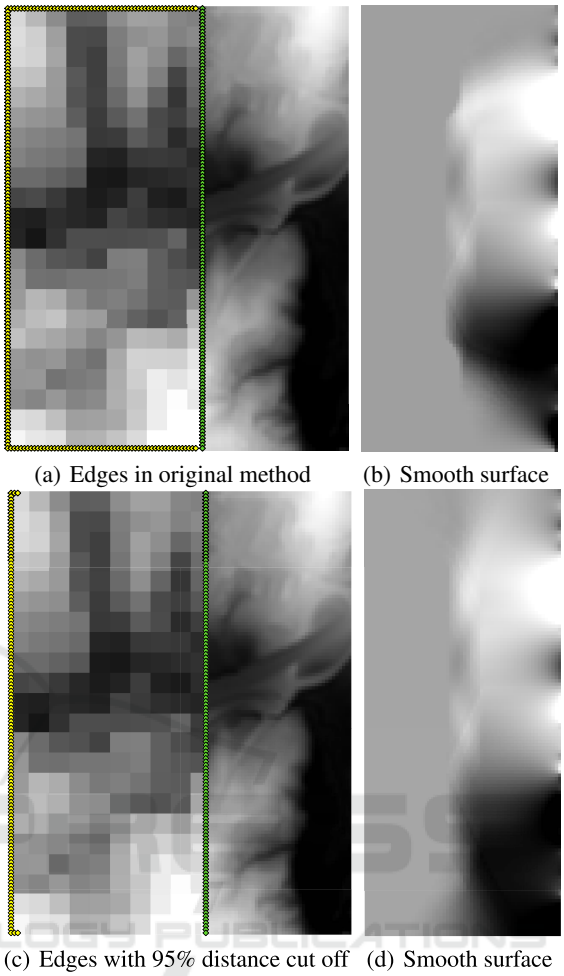
The main difference from the `r.mblend` implementation to the original method concerns the computation of the *far edge*. `r.mblend` uses by default only those points in the *far edge* that are farther away from the *near edge*, with the aim of obtaining a geometrically even transition in the smoothing surface. In detail, this computation is performed by `r.mblend` as follows:

1. Compute a distance map within the low resolution area relative to the high resolution raster.
2. Vectorise the distance raster into a points data set.
3. Normalise the distance values and select those above a certain threshold (by default 95% of the maximum distance).
4. Use an inner buffer to the interpolation area to select further those points only along the low resolution raster border.

Figure 1 presents these differences to the original proposal with a simple case. The user is able to adjust the distance cut off thus dosing the weight of the *far edge* on the smoothing surface interpolation.

The `r.mblend` implementation also provides the user with the option to use the average difference between the two input DEMs as the value assigned to the *far edge* interpolation points (instead only zero, as in the original proposal). In this mode the resulting DEM remains closer to the high resolution input. This may be useful when the differences between the two DEMs are spatially uncorrelated.

[2]http://www.numpy.org/
[3]http://pandas.pydata.org/
[4]https://github.com/ldesousa/r.mblend
[5]http://ec.europa.eu/idabc/eupl.html



(a) Edges in original method (b) Smooth surface

(c) Edges with 95% distance cut off (d) Smooth surface

Figure 1: Interpolation points edges with original method (a) and a 95% cut off to the maximum distance (c) and the respective smoothing surfaces (b and d). *Near edge* in green, *far edge* in yellow.

3.3 Model Parameters

The `r.mblend` module takes the following arguments:

- `high` - name of the high resolution DEM;
- `low` - name of the low resolution DEM (overlapped by the high resolution input);
- `output` - name of the resulting blended DEM;
- `far_edge` - percentage of the maximum distance to the high resolution DEM used to determine the *far edge*;
- `inter_points` - number of points (from both edges) to use in interpolation;
- `-a` - optional flag that indicates to assign the average difference between the two input rasters to the *far edge* (instead of zero).

The far_edge argument is bounded between 0 and 100; by default is used a value of 95. Values closer to zero translate into a higher number of points in the far edge, impacting the shape of the differences raster.

The Inverse Distance Weighting (IDW) method is used to interpolate the smoothing surface. This method takes as a parameter the number of points (from the two edges) used to interpolate each new cell value. By default 50 points are used; inter_points provides the user a mean to tweak this value. The higher the number of points used the smoother is the resulting smoothing surface; however, it also means a lengthier computation time.

4 TEST CASES

4.1 A - Lucern

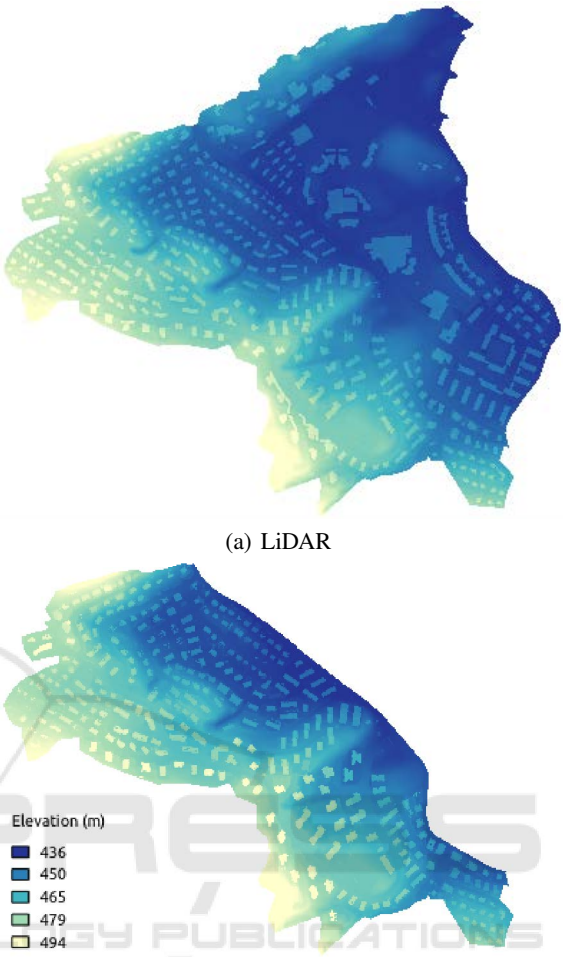
For the first test were employed two DEMs representing an urban catchment in the city of Lucerne in Switzerland. This is a relatively smooth surface, but including a number of detailed man-made features. The lower resolution DEM was obtained with an air-borne LiDAR sensor and provided to this study by the official cadastral service of the Canton of Lucerne. It has a cell side of 0.5 metres and a vertical accuracy of approximately 0.5 metres (Figure 2a). This dataset was last updated in July of 2012 (Doe, 2014).

The high resolution DEM was obtained with a conventional camera mounted aboard an electricity powered, fixed-wing UAV. This UAV made several flights at an altitude of 114 metres over the study area in March of 2014. Overlapping images where acquired from different angles allowing for stereoscopic depth rendition. The resulting DEM has a spatial resolution of 0.5 metres and a vertical accuracy of 0.2 metres (Figure 2b).

4.2 B - North Carolina

The second test case was derived from the open spatial data set from North Carolina distributed with GRASS as sample data [6]. This data set includes a 10 metres cell side DEM representing relatively rugged terrain with carved valleys and sparse man-made features. A section of this DEM was cropped to be used as high resolution input (Figure 3b). The original DEM was then converted to a lower spatial resolution with 60 metres side cells, to which non spatially correlated noise was added (Figure 3a).

[6] https://grass.osgeo.org/download/sample-data/



(a) LiDAR

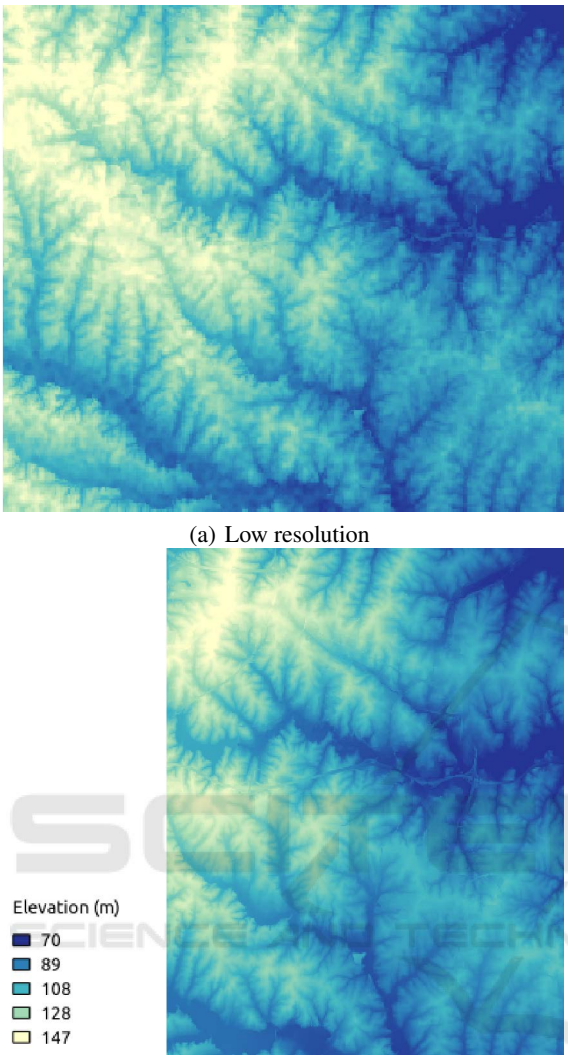
(b) UAV stereoscopic

Figure 2: Overlapping DEMs of different resolutions built from the North Carolina data set.

5 RESULTS

To compare r.mblend with the *Cover* and *Average* methods the Mosaic tool provided with the ArcGIS software was used. This is tool is able to merge DEMs using both types of conventional methods. All results where then assessed with GRASS.

A high pass filter was used for a first assessment of the merged DEMs produced by each of the three methods. A 5-by-5 cell filter was used in order to highlight zones of transition, e.g. sharp edges, walls and so forth. Figure 4 presents a detail of these results for the Lucerne test case. Immediately standing out is the artificial step introduced by the *Cover* and *Average* methods along the border between the two input DEMs. The step is not so marked with *Average*, but still present; contrariwise, at a closer inspection a loss of detail is visible with this method, with various



(a) Low resolution

(b) High resolution

Figure 3: Overlapping DEMs of different resolutions built from the North Carolina data set.

small transitions in the UAV DEM loosing magnitude. As for `r.mblend` it shows no border at all, while preserving the fine detail in the high resolution DEM.

In the North Carolina test case the artificial step introduced by the *Cover* and *Average* methods is also present, even if less marked (Figure 5). In this case the transitions within the larger cell areas stand out considerably more. Since this is rugged terrain, the 60 metres cells introduce relevant ridges and cliffs. It is also interesting to observe the effects of the *Average* method on the high resolution area, introducing the artificial ridges from the 60 meters DEM.

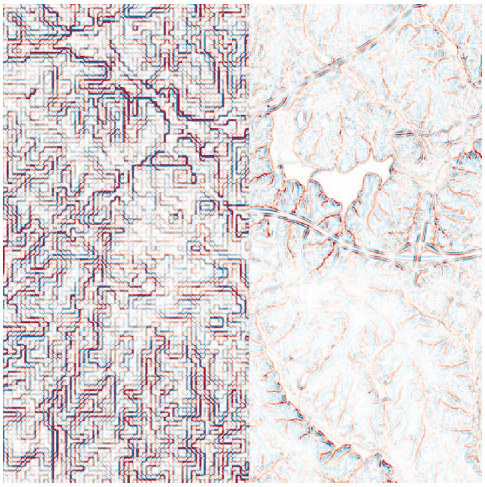
Taking the differences from the original to the output DEMs provides another point of assessment. Figure 6 shows together the differences from the blended result to the high resolution DEM and the dif-

(a) *Cover* method

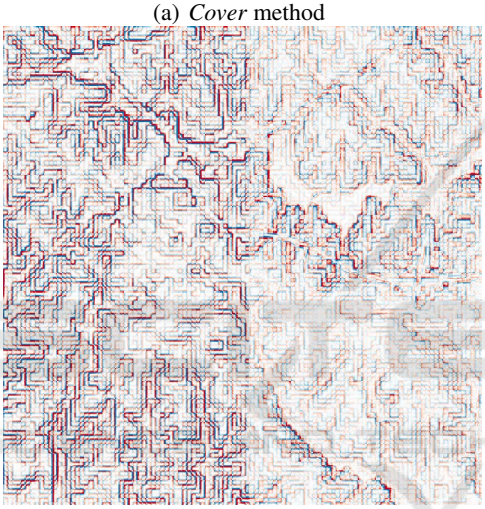
(b) *Average* method

(c) `r.mblend`

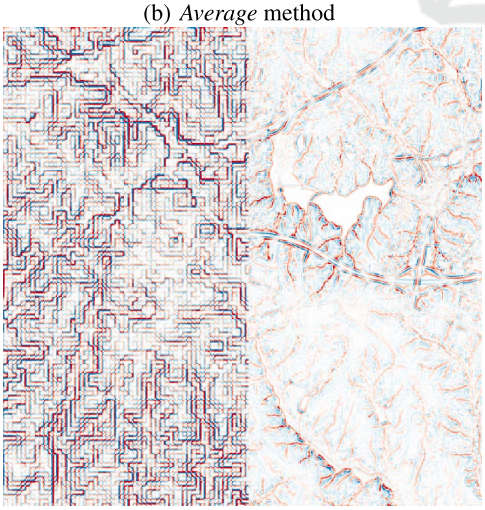
Figure 4: Results of high pass filter applied on merged DEMs in test case A.



(a) *Cover* method



(b) *Average* method

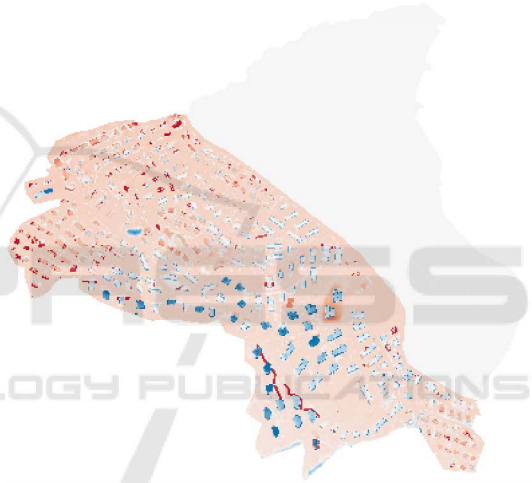


(c) r.mblend

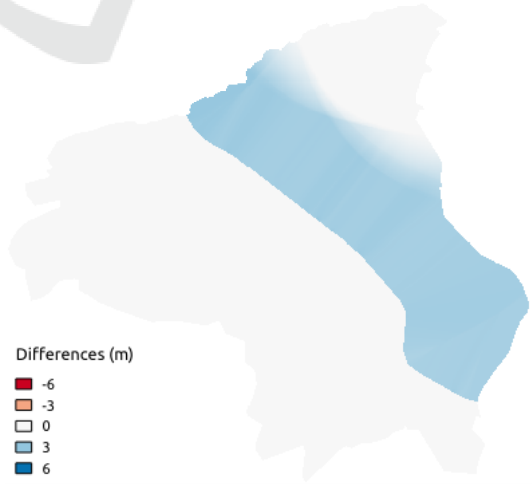
Figure 5: Results of a high pass filter applied on merged rasters in test case B.

ferences from the result to the low resolution DEM in the areas not covered by the high resolution input. This analysis is not presented for the *Cover* method since it does not change the inputs. The differences between *Average* and r.mblend are striking, appearing in opposite areas. r.mblend applies changes only to the low resolution DEM, with a smooth transition surface; *Average* leaves the low resolution data untouched, while applying irregular and many times severe changes to the high resolution data.

A similar pattern in differences is patent in the North Carolina test (Figure 7). r.mblend yields again the smooth transition surface, applying changes solely to the low resolution input. As before, the *Average* method introduces broad changes to the area where both inputs overlap, in this case coinciding with the full extent of the high resolution DEM.

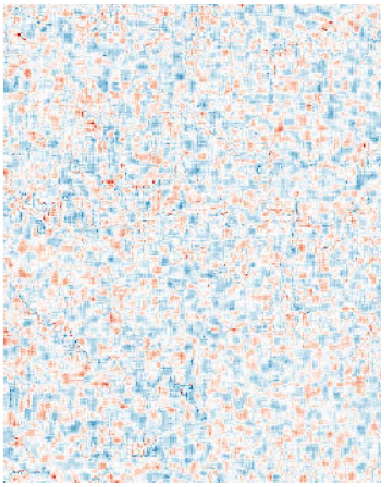


(a) *Average* method

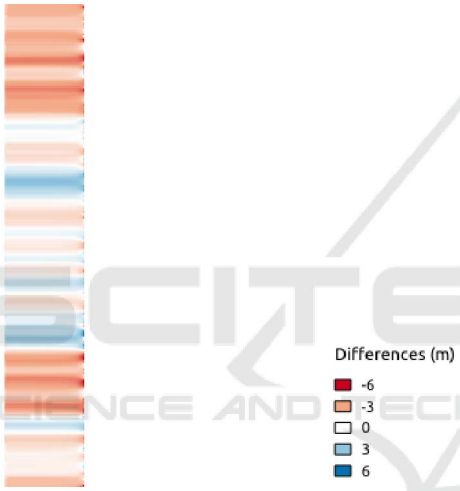


(b) r.mblend

Figure 6: Differences from resulting blended DEM to inputs in test case A.



(a) *Average* method



(b) `r.mblend`

Figure 7: Differences from resulting blended DEM to inputs in test case B.

6 SUMMARY AND FUTURE WORK

This article compared the `r.mblend` GRASS addon with conventional methods to merge overlapping DEMs of different spatial resolution. Using two different test cases, it was possible to assess its advantages on smooth and rugged terrain. `r.mblend` eliminates the steps introduced along the border of the areas where the merging inputs overlap; these steps are not so marked in rugged but still present. This smooth transition is not achieved at the expense of loss of detail, as the high resolution DEM is left untouched. This contrasts particularly with the *Average* method, that visibly derides the information from the high resolution input. `r.mblend` presents itself as clearly superior alternative to the conventional methods assessed.

Presently, `r.mblend` operates on a single execution thread. All operations conducted are relatively straightforward, except for the interpolation of the smoothing surface. For the Lucern case study presented above, this operation may take in the order of dozens of minutes. However, it is possible to parallelise this operation, since there is no dependence between cells of the resulting surface. The GRASS Python API provides elementary tools for parallelisation, spawning GRASS commands as sub-processes. Therefore, an obvious evolution to `r.mblend` is to slice the interpolation area and run the interpolation independently on each slice.

Other ways of improvement also concern the smoothing surface interpolation. Alternative methods beyond IDW can be made available to the user, as so their respective parameters. This would provide the user with further degrees of freedom to tune the module output.

Finally, `r.mblend` can also be extended to automatically apply an high pass filter on the resulting DEM, providing it as a secondary output. This is a useful asset to assess to quality of the resulting DEM, either visually or in more elaborate analysis.

REFERENCES

de Smith, M. J., Goodchild, M. F., and Longley, P. A. (2015). *Geospatial Analysis: A Comprehensive Guide to Principles, Techniques and Software Tools - Fifth Edition*, chapter Geocomputational methods and modeling, pages 625–672. Winchelsea Press.

Doe, R. (2014). GIS Kanton Luzern. `https://rawi.lu.ch/themen/gis_kanton_luzern`. Accessed: 30-10-2014.

Eastman, J. (2012). *IDRISI Selva*. Clark University, MA, USA.

Hickey, R. (2000). Slope angle and slope length solutions for GIS. *Cartography*, 29(1):1–8.

Katzil, Y. and Doytsher, Y. (2005). Spatial rubber sheeting of dtms. In *Proceedings of the 6th Geomatic Week Conference, Barcelona, Spain*, volume 811.

Küng, O., Strecha, C., Beyeler, A., Zufferey, J.-C., Floreano, D., Fua, P., and Gervaix, F. (2011). The accuracy of automatic photogrammetric techniques on ultralight uav imagery. In *UAV-g 2011-Unmanned Aerial Vehicle in Geomatics*, number EPFL-CONF-168806.

Leitão, J., Prodanović, D., and Maksimović, Č. (2016). Improving merge methods for grid-based digital elevation models. *Computers & Geosciences*, 88:115 – 131.

Luedeling, E., Siebert, S., and Buerkert, A. (2007). Filling the voids in the srtm elevation modela tin-based delta

surface approach. *ISPRS Journal of Photogrammetry and Remote Sensing*, 62(4):283–294.

Neteler, M., Bowman, M. H., Landa, M., and Metz, M. (2012). Grass gis: A multi-purpose open source gis. *Environmental Modelling & Software*, 31:124–130.

Sanner, M. F. et al. (1999). Python: a programming language for software integration and development. *J Mol Graph Model*, 17(1):57–61.