A FRAMEWORK FOR AN AUTOMATIC SEAMLINE ENGINE

M. Al-Durgham [a], M. Downey [b], S. Gehrke [c], B. T. Beshah [b]

[a] aldurgham@gmail.com
[b] North West Geomatics Ltd., 245 Aero Way NE, Calgary, AB T2E 6K2, Canada – {michael.downey | belai.beshah}@nwgeo.com
[c] Hexagon Geosystems, Leica Geospatial Solutions Division, Goethestr. 42, 10625 Berlin, Germany – stephan.gehrke@hexagongeosystems.com

Commission I, WG I/4

KEY WORDS: Aerial Images, Orthophoto, Mosaic, Radiometry, Seamlines, Graphcut

ABSTRACT:

Seamline generation is a crucial last step in the ortho-image mosaicking process. In particular, it is required to convolute residual geometric and radiometric imperfections that stem from various sources. In particular, temporal differences in the acquired data will cause the scene content and illumination conditions to vary. These variations can be modelled successfully. However, one is left with micro-differences that do need to be considered in seamline generation. Another cause of discrepancies originates from the rectification surface as it will not model the actual terrain and especially human-made objects perfectly. Quality of the image orientation will also contribute to the overall differences between adjacent ortho-rectified images.
Our approach takes into consideration the aforementioned differences in designing a seamline engine. We have identified the following essential behaviours of the seamline in our engine: 1) Seamlines must pass through the path of least resistance, i.e., overlap areas with low radiometric differences. 2) Seamlines must not intersect with breaklines as that will lead to visible geometric artefacts. And finally, 3), shorter seamlines are generally favourable; they also result in faster operator review and, where necessary, interactive editing cycles. The engine design also permits alteration of the above rules for special cases.
Although our preliminary experiments are geared towards line imaging systems (i.e., the Leica ADS family), our seamline engine remains sensor agnostic. Hence, our design is capable of mosaicking images from various sources with minimal effort. The main idea behind this engine is using graph cuts which, in spirit, is based of the max-flow min-cut theory. The main advantage of using graph cuts theory is that the generated solution is global in the energy minimization sense. In addition, graph cuts allows for a highly scalable design where a set of rules contribute towards a cost function which, in turn, influences the path of minimum resistance for the seamlines. In this paper, the authors present an approach for achieving quality seamlines relatively quickly and with emphasis on generating truly seamless ortho-mosaics.

1. PREREQUISITS

Before we dive into the details of the seamline engine, it is worth discussing the existing technologies utilized by our algorithm. In this section, a brief description of the watershed segmentation and graph cut theory is presented.

1.1 Watershed Segmentation

The watershed by flooding (Beucher S., and Lantuejoul C., 1979) algorithm is – conceptually – based on how water flows into areas of local minima and form separated pools. We can treat any grey scale image as a topographical surface where bright areas are seen as high and dark areas as low allowing the watershed algorithm to segment image into pools. The boundary of each pool is the local maxima, and as presented in this paper, we can utilize the watershed segmentation to identify a network of possible seamlines from a cost image. Although other variations of the watershed segmentation algorithms exist (for example in Cousty J., et al., 2009), the basis of a watershed segmentation used in this work can be described in the following steps:

1. Identify a set of markers within an image that represent local low points. These local minimas serve as the seeds for the watershed algorithm. Hence, it is crucial that these points are selected carefully.
2. Starting with the lowest unmarked minima next to a segment, a pixel flooding operator marks pixels with the same value as their neighbouring seed.
3. If the pixel falls between multiple watershed neighbours then this is a boundary pixel and hence it is marked as a barrier.
4. The above operations are repeated until all pixels have been flooded or identified as a barrier.

As can be seen from the above enumerations, the watershed algorithm is fairly simple and does not require a significant number of operations. The main challenge is in choosing the markers carefully. The authors offer an automatic approach to pick the markers that works well in the context of seamline generation as a presented in the text later on.

1.2 Graph Cuts

Graph cuts (Boykov Y., and Funka-Lea G., 2006) is a classification algorithm that can be used in a wide range of vision applications including the binary segmentation of images. To classify a scene using graph cuts, one has to build a graph first. Typically, the nodes of the graph correspond to image segments. Weighted connections between the nodes can be established to reflect the strength of connectivity between the nodes. The graph is then connected to a source and sink nodes and weights are also introduced between the source/sink and the rest of the graph nodes.

Figure 1 (left) shows a conceptual representation of a graph for binary classification. The blue dots are the nodes to be segmented, the blue lines are weighted connections between the nodes. The red dot is the source node with connections to the graph nodes, while the green dot represents the sink node that is also connected to the nodes. Since weighting is incorporated in the connections, a node can be connected to the source and sink at the same time. Also, a node can be connected to neither the sink nor the source. The classification of such node (into sink or source) depends on the connections of that particular node to its neighbours. Figure 1 (right) depicts the results of classification and the "cut" line represents the border between source and sink objects (i.e., the seamline in our case).

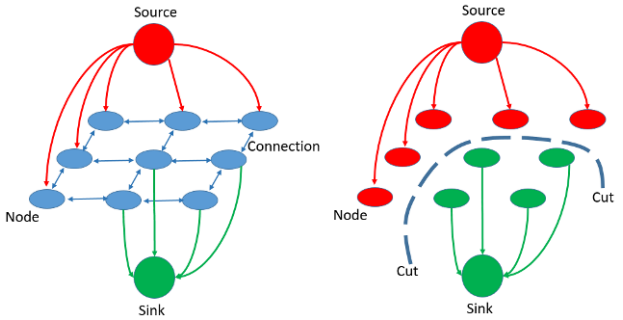

Figure 1. A representation of a graph (left) and the resulting cut (right), Figure taken with adaptation from (Boykov Y., and Funka-Lea G., 2006)

The weighting mechanism of the connections throughout a graph greatly influences the maximum flow (i.e., minimum cut) path. Hence, we use the local pixel differences at the segments boundary to influence the strength of the connections. The higher the weight the less likely that the graph cuts solution will pass between two segments.
Generally speaking, the higher the overlap between overlapping images, the higher the number of nodes in the graph and hence, the better the odds of finding a good path for the seamlines. However, it is possible that the watershed segmentation leads to large segments that spans all the way across the image overlap. In this case, that segment must be broken into multiple pieces to ensure a proper graph cut solution.

2. SEAMLINE ENGINE WORKFLOW

To better demonstrate the seamline engine, one could start with a simplified example consisting of the image pair depicted in Figure 2. These images have been geometrically orthorectified and radiometrically balanced using a new radiometric normalization method (Gehrke S., and Beshah B., 2016). In addition, the exterior (eop) and interior orientation parameters (iop) are preserved for a later stage in the process. A digital surface model (dsm) is also computed from the original image geometry (Gehrke S., et al., 2011) or using an external source such as lidar. The dsm is optional and – when available – can be used to identify breaklines in the scene. These breaklines can be used as constraints to influence the path of the seamlines to pass around them which improves the quality of the generated seamlines.
We start by utilizing an absolute difference operator where the two orthophotos are subtracted to generate one image. The size of this image is the minimum bounding rectangle of the overlap area. The difference image is computed for each band and is intended to highlight discrepancies due to image mismatch (i.e., features lean in different directions, reflective surfaces, reflections, cloud patches, etc.)
Next, the differences are inverted and normalized such that large radiometric differences correspond to the darkest areas in the difference image. I.e., the areas where there is virtually no difference will appear in the difference image as white. This image is referred to in the rest of the text as the constraint mask. Following steps will compound on top of this image which defines areas where seamline pass-through is discouraged. It is worth reminding the reader here, that the constraint mask is not a binary image but rather greyscale which allows for better influence on the seamline path. The greyscale values can also be manipulated to weigh constraints proportional to their severity as shown in the equation below. Where, a_i is a scale factor that dictates the contribution of a given constraint c_i towards the constraint mask.

$$c = \sum_{i}^{x} a_i c_i \quad , where \sum_{i}^{x} a_i = 1$$

Figure 2. An example depicting two overlapping orthophotos (top image is left in the overlap and the bottom image is right)

Figure 3. A constraint mask derived from the image differences

At this stage, the contributions from a height source – shown in Figure 4 – towards the constraint mask are added. A gradient operator is executed over the dsm to produce a slope graph. The values of the detected edges (e.g., buildings boundary, bridges, and isolated trees) are inverted, normalized, and added to the constraint mask. These constraints play a crucial role in preventing seamlines from crossing through building rooftops hence reducing the possibility of seamline artefacts.

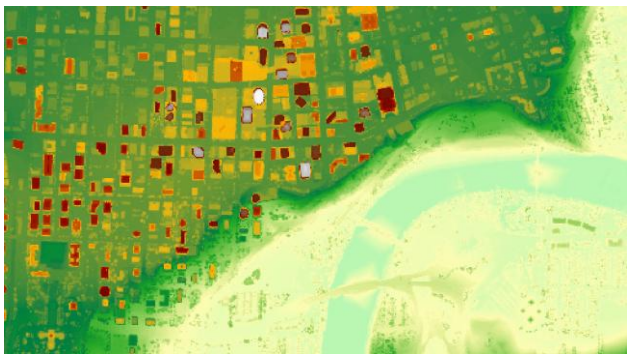

Figure 4. A colour coded digital surface model (red represents high elevations and light blue represents low elevations)

A median filter is then used to remove speckles from the constraint image. The speckles are mainly caused by small objects in the scene. Furthermore, dilation and erosion filters are performed to connect nearby constraint features. This will help reduces the number of watershed segments later on.

Figure 5. A constraint mask after adding breakline contributions

Generally speaking, it is desired that a seamline passes through the centre of the overlap area. This is preferred to reduce the discrepancy in objects lean on both sides of the seamline. To encourage such behaviour, we introduce an artificial surface (Figure 6) that gradually penalize the overlap pixels as they move away (i.e., left and right) from the overlap centre. Scale factor (**Error! Reference source not found.**) associated with this constraint is set much lower than the other constraints to prevent this pattern from dictating the seamline path. This gradual mask also serves another crucial purpose, it prevents the seamline from terminating prematurely at either side of the overlap and ensures that the seamline extends throughout the entire overlap area.

Using the dsm and the sensor model, one can compute areas in the overlap space that result from sharp intersection angle between the image rays and the surface normal direction. These areas appear usually as smears in the orhtophoto product since fewer pixels in the original image are stretched over many pixels in the digital elevation model. These patches can be detected by analysing the angle between the image-ray and the surface-normal vectors.

Since one has no control over the imagery content, our algorithm also allows operator to introduce an optional mask that may highlight other artefacts detected during the image quality control checks.

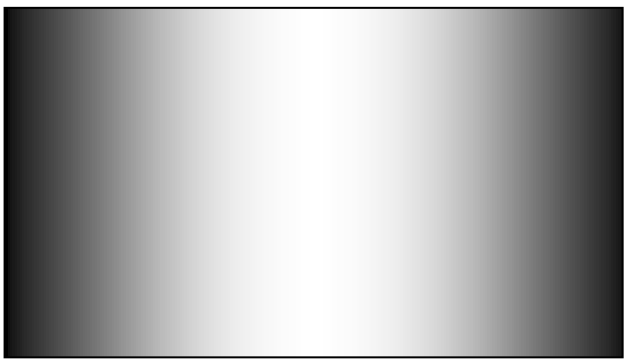

Figure 6. A conceptual representation of the gradient mask

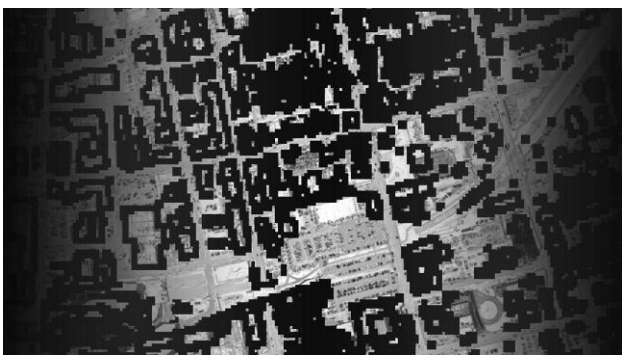

Figure 7. A conceptual representation of final constraint mask

One could think of the constraint mask as an energy field. Bright areas correspond to low resistance (hence high flow) pixels and darker areas correspond to high resistance pixels. At this stage, one could search all the pixels to find the path of minimum resistance. However, due to the sheer volume of pixels (i.e., in the order of tens of millions of pixels for a typical overlap), it is better to cluster the pixels into groups.

Hence, a watershed kernel is performed over the constraint mask. The watershed operator will segment the constraint mask into patches surrounding the highest constrained pixels. Figure 8 depicts the watershed segmentation boundary superimposed over one of the sample image.

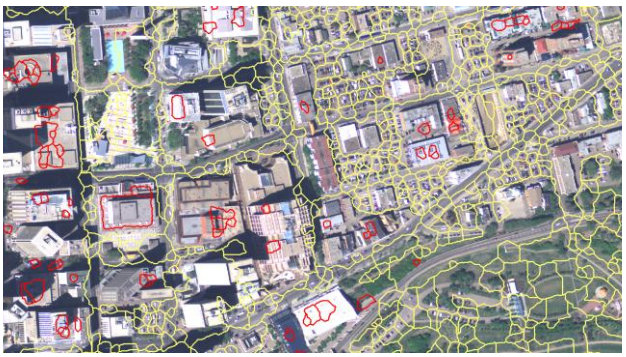

Figure 8. A colour coded image depicting a segmented constraint mask

As can be seen in this figure, there are some watershed segments residing inside bigger watershed segments. These segments shown in red referred to as "islands" and ignored by diluting them into their enclosing segments. The remaining segments (shown in yellow) constitute a network of possible seamlines and the only remaining task becomes to identify the most efficient route for seamlines.

To define the best possible seamline route, we have first to discuss the factors that make a seamline seamless. 1) The pixels on each side of the seamline should be homogeneous (i.e., very similar in colour and texture). Hence when a seamline passes through it won't be noticed. Furthermore, pixel blending around the seamline is less likely to cause artefacts if the surroundings are colour-homogeneous. And 2) Shorter seamlines are preferred. Without this constraint, seamlines could "wonder" in overlap patches that depict little resistance (e.g., a grass field, calm lake, etc.). This condition also helps in saving QC operations time by reducing the seamline length.

To find the best path, we build a graph where the nodes are the watershed patches and the connections between the nodes represent costs derived from the number of shared pixels between any two neighbouring patches. In addition, the actual pixel values (from the constraint mask) along the two patches are aggregated and added to the connection cost. Figure 9 shows an example of the nodes and connections derived from the watershed segments.

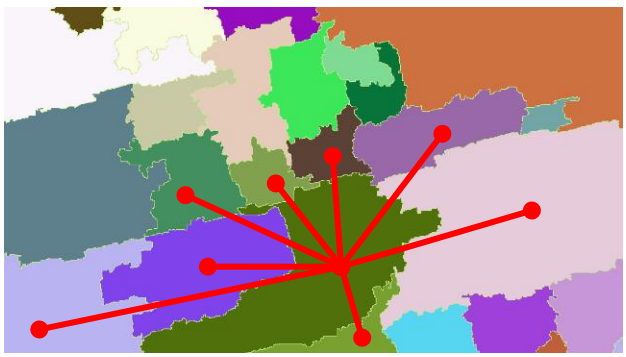

Figure 9. A representation of the nodes (red dots) and connections (red lines) derived from watershed segments

Once all the connections between the adjacent patches are realized, the graph is almost complete. The only remaining connections are those linking the graph to the source and sink. The choice of these connections determines in which direction (e.g., top-down vs. right-left) the cut (i.e, the seamline) will flow. To resolve this, strong connections are set between the overlap boundary segments and the source/sink nodes. This way one is confident that the graph will be split in the desired general direction. Finally, a max-flow algorithm (Boykov Y., and Kolmogorov V., 2004) is used to define min-cut in the graph. Hence, the graph nodes are now classified to either belong to the "source" or the "sink" nodes. The border line between these two groups resembles the desired seamline.

Figure 10 shows a sample result of the seamlines generated via graph cuts. The line between the left (green) and right side (red) is the seamline. Notice how the seamline successfully avoids breaklines in building structures.

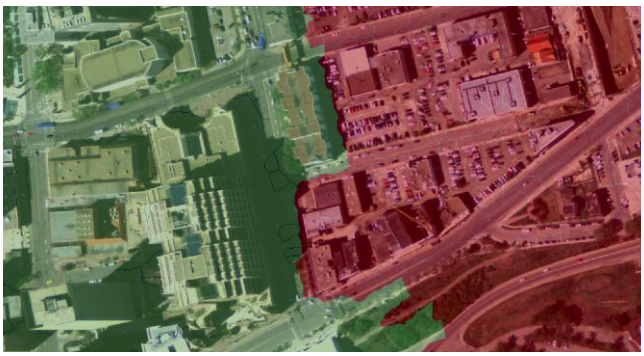

Figure 10. An example of the graph-cut results superimposed on the left image

Figure 11(a-c) depict few examples of seamline results around human-made objects as well as natural barriers (i.e., canopy lines). It can be seen from these examples that the constraint mask works well in influencing the seamline behaviour. In these images, the pixels on the left of the seamline belong to the left image and the ones to the right of the seamline belong to the right image. Note also that the results are shown without feathering or pixel blending for the pixels closest to the seamlines. More seamline examples are also shown in Figure 14 and Figure 15

Figure 11.(a) An example seamline dodging a road sign

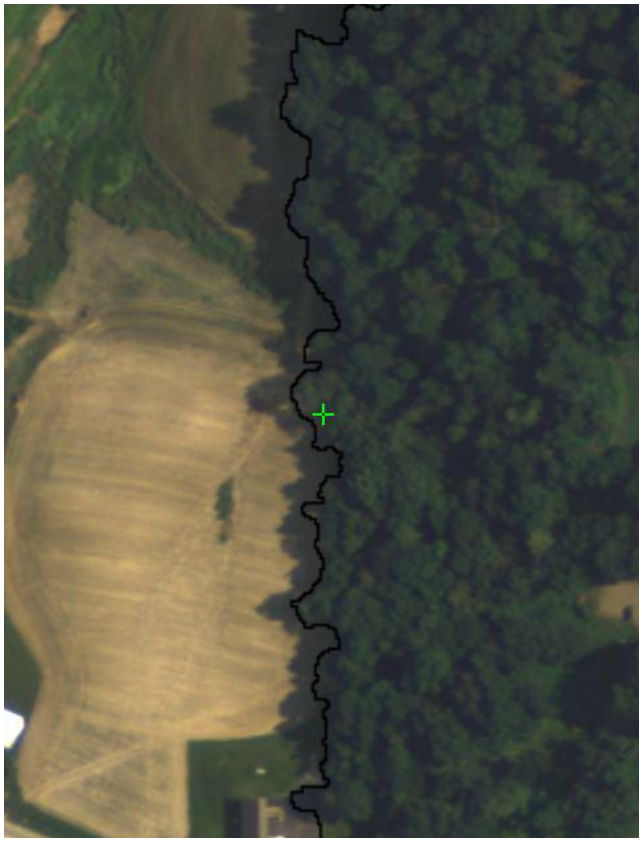

Figure 11.(b) A seamline passing the shadow line between a canopy and a fie

Figure 11.(c) A seamline example passing next to a bridge

Since it is clear now how a seamline can be generated for a pair of images, the concept can be generalized to mosaic many more images. This can be achieved by introducing the concept of a "virtual image". A virtual image is made of two or more images that has been seamlined together and are treated as a single image. A virtual image can also be combined with another virtual image using the aforementioned pairwise seamline engine to generate a bigger virtual images as shown in Figure 12. Using this concept, one can construct a tree that combines many images into the desired final mosaic similar to Figure 16.

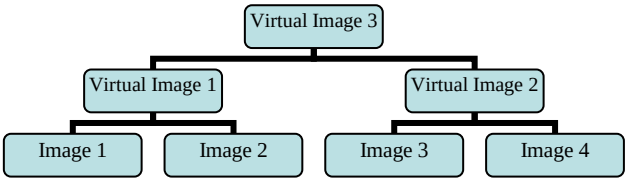

Figure 12. An example depicting the generation of a large image mosaic using many images

In the above figure, one can see that there are dependency between consecutive pairwise operations. Although true, this algorithm runs efficiently in a cluster environment with minimal impact on the algorithm's level of parallelism. Clever ordering of the pairwise jobs combined with implementation tricks can limit the processing dependency. For example, assuming that Images 1-4 in Figure 12 have 60% forward lap with its neighbours, one does not have to wait for "virtual image 1" and "virtual image 2" to finish before processing "Virtual image 3". Instead, all three processes can be executed simultaneously and as the pairwise regions needed for processing are fairly separate (i.e., the 10% overlap between images 1 and 3 for example is ignored). Figure 13 summarizes the seamline generation algorithm through a workflow diagram.

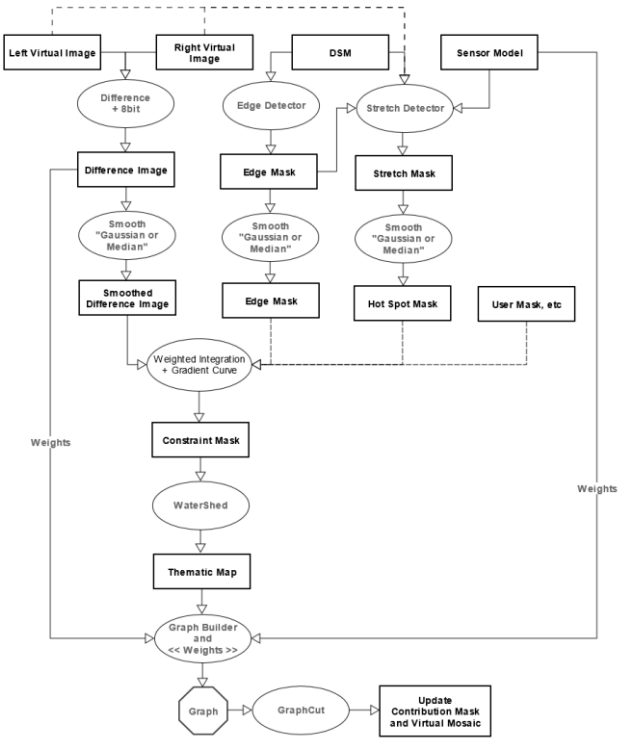

Figure 13. A workflow diagram of the seamline engine

Figure 14. An example of a seamline following running across a road

Figure 15. An example of a seamline following a curved road

3. CONCLUDING REMARKS

In this short paper, the authors presented their approach for solving the geometric mosaicking problem of ortho-rectified imagery. Using watershed segments that are derived from a constraint image, a network of possible paths is constructed. Any segment of this network serve as the best seamline route within its locale. A network based of watershed segments can be converted into a weighted graph where the nodes represent the watershed segments and the connections between the segments are weighted based on the length and values of the pixels falling at the boundary between the segments. A global minimization can be achieved by connecting source and sink nodes then performing graph cut. The result of the graph cuts classifies every node into either source or sink, the border between the source and sink nodes is the desired seamline.

The authors have shown that the above procedure can be generalized to mosaic many images by introducing the virtual image concept. The seamline results in this paper are presented without seamline simplification, feathering, or pixel blending which remains a subject of future research and development.

Figure 16. Four flight lines grouped into a seamless mosaic, the yellow lines are the footprints of the images (ADS100 sensor)

REFERENCES

Beucher S., Lantuéj C., 1979: Use of Watersheds in Contour Detection. International Workshop on Image Processing: Real-time Edge and Motion Detection/Estimation, Rennes, France.

Boykov Y., and Funka-Lea G., 2006: Graph cuts and efficient ND image segmentation. International journal of computer vision, 70(2), pp.109-131.

Boykov Y., and Kolmogorov V., 2004: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 26(9), pp.1124-1137.

Cousty J., Bertrand G., Najman L., and Couprie M., 2009: Watershed Cuts: Minimum Spanning Forests and the Drop of Water Principle in Pattern Analysis and Machine Intelligence, IEEE Transactions on 31(8) pp. 1362-1374.

Gehrke S., and Beshah B., 2016: Radiometric Normalization of Large Airborne Image Data Sets Acquired by Different Sensor Types, XXIII ISPRS Congress, Prague Czech Republic (submitted).

Gehrke S., Uebbing R., Downey M., and Morin, K., 2011: Creating and using very high density point clouds derived from ADS imagery. Proceedings of the American Society of Photogrammetry and Remote Sensing 2011 Annual Conference, Milwaukee, WI, USA (Vol. 15).