TECHNIQUES FOR THE GRAPH REPRESENTATION OF SPECTRAL IMAGERY

Ryan A. Mercovich (ram4238@cis.rit.edu), James Albano, David Messinger

Center for Imaging Science, Rochester Institute of Technology

ABSTRACT

Many techniques from graph theory and network theory have been applied to traditional images, and some techniques are now being applied to spectral imagery. Contrary to the typical approaches of utilizing the first order statistics, mixture models, and linear subspaces, the methods described in this paper utilize the spectral data structure to generate a graph representation of the image. By ignoring any reliance on the shape of the data, graph based methods can succeed where typical methods break down, such as in high resolution scenes with very high clutter. Before graph theory techniques can be utilized on an image, it must be represented as a graph. Because images contain only measured nodes (pixels) and no edges, edges are drawn between pixels based on some similarity measure. With a specific focus on creating graphs for clustering, several graph creation techniques are compared with two novel methods described: the locally weighted k-nearest neighbor approach and the density weighted k-nearest neighbor approach. By applying two different clustering techniques to the resulting graphs, the various graph creation techniques are compared using real world data.

Index Terms— graph theory, data representation, spectral clustering, hyperspectral

1. INTRODUCTION

Recently techniques from graph theory have been applied to spectral image processing [1,2,3,4]. Additionally, similar techniques have been used in traditional image processing [5, 6, 7]. To utilize graph theory, the image data must be represented as a graph. Graph representation is most effective when traditional data models, such as first and second order statistics, linear mixture models, or linear subspaces, do not represent the data at hand. This separation between traditional data models and spectral image data increases for high resolution aerial and satellite imagery that generally contains more clutter than lower resolution sensors.

A graph, consisting of a set of $nodes$ and a set of $edges$, is a labeling of data points and connections between those points. To represent an image with a graph, the image must be examined in the *n*-dimensional space spectral space. Each pixel becomes a node, and edges can be drawn between pixels with a certain similarity or connectedness. Once the pairwise adjacencies are defined, the graph can be represented as an $adjacency\ matrix$, and many techniques can be applied to this matrix to help identify community structure [5, 8], define the dimensionality of the data[4,7], or even for anomaly and target detection[1,4].

Graphs are well studied in the mathematics world, and are widely applied to network theory problems [9]. For imaging, it is important to know that the graph is not a part of the measured data, it is created based on observed similarity between pixels. However, once created, the graph can be treated just like it is for other disciplines. The graph for a set of x pixels in n dimensions is represented by an $x \times x$ adjacency matrix. Each pixel has a certain $degree$, defined by the number of edges incident with it, and each edge can have a weight.

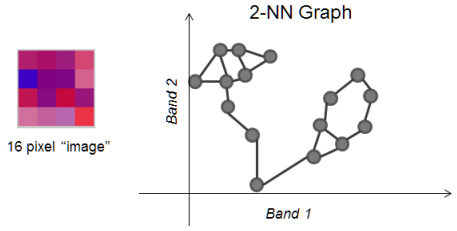

Fig. 1. An image and its associated graph. Each pixel is represented by a node. Every node is connected to its two nearest neighbors (2-NN) in a Euclidean sense.

Graphs are a useful way to represent image data because of the large amount of information that can be gleaned from them. Rather than the identification of the middle of (mean or median) and measuring the spread of (covariance) the distribution, a graph provides a much more detailed model for the data. In addition, the graph can be recursively split for automatic clustering [2,3,5]. Many techniques for identifying community structure in a graph exist in the literature [10,11,12]. The key to applying these techniques to imaging is the proper creation of the graph. Images are represented as graphs to improve the description of the data structure, and the process of creating the graph must be undertaken carefully to avoid injecting data structure that is not truly present.

As stated previously, graphs are utilized when traditional models insufficiently characterize the data. In some imaging regimes, traditional models break down [13]. The amount of image clutter, a small ground sample distance, and other characteristics can be used to determine if a graph representation will be useful for a specific situation.

2. METHODS

2.1 Background

The process of representing an image as a graph can be quite straightforward. The most simple technique, and one used often for high resolution gray scale or RGB imagery, is to connect pixels based on their spatial proximity. This is known as a simple grid graph [6] and is shown in **Figure 2**.

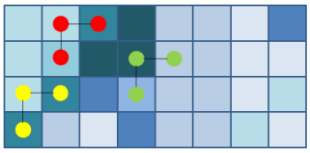

Fig. 2. An example construction for a grid graph for the three pixels highlighted. Adjacency is assigned based on spatial relationship instead of spectra (here adjacencies are made with the right and lower neighbors).

A more advanced but still straightforward method is to use a distance metric, such as the Euclidean distance, in the spectral space to connect a pixel to its k nearest neighbors (kNN, as seen in **Fig. 1.**). Another distance based technique would involve connecting all pixels within a certain distance, d [6]. Of many possible distance metrics, the Euclidean distance and spectral angle are used here and in the following techniques.

The process of exhaustively searching for neighbors based on distance becomes increasingly time-consuming as the dataset increases in size. To perform this search more efficiently, methods have been developed for fast nearest neighbor search [14]. One such method, developed to utilize several different distance metrics is known as ATRIA. The fast nearest neighbor search using the ATRIA tree structure is well known and straightforward to implement. The nearest neighbor search is accelerated by utilizing a tree structure to limit the search for neighbors.

2.2 Locally weighted nearest neighbors

One problem with the kNN techniques is that the graphs produced are often not fully connected. Connected graphs are those in which a path (along edges) exists from any point n to any other point m and are desirable for many graph theory clustering methods [5, 7, 10]. Grid graphs are fully connected by design. One way to ensure a connected graph is to create a *spanning tree* in the data in addition to the edges defining the community structure. The spanning tree is a graph where every node is connected, but no cycles (or loops) exist. This process can be computationally expensive if done in a way that does not alter the graph's community structure (*i.e.* the minimum spanning tree).

To combat the connectivity problem without adding additional compute time, a new method combining the kNN and grid techniques is proposed. For each pixel the first l neighbors can be forced to come from the same spatial region as that pixel. For example, when 25 neighbors are desired for a pixel x, the first 5 (k_l) might come from the 30 nearest pixels in a spatial sense, while the remaining 20 come from the global dataset.

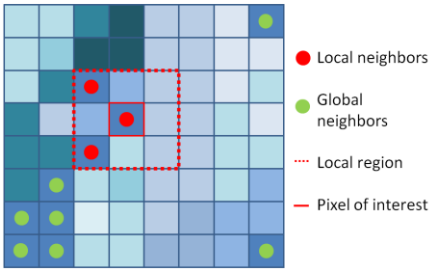

Fig. 3. The locally weighted method takes spectrally similar pixels from the local neighborhood and the global data. Forcing some pixels from the local neighborhood improves graph connectivity.

In addition to forcing the local neighbors to improve connectivity, the distance metric can be adjusted to improve adjacencies. The local neighbors can be identified using the spectral angle. The spectral angle can be a poor metric for finding global neighbors because of its performance when applied to very dark spectra. However, for pixels in the same small spatial region the spectral angle can help to find adjacencies for pixels in and out of shadow. The local weighting can also be applied to the following techniques.

2.3 Mutual k-nearest neighbors

The mutual kNN method is similar to the kNN method but instead creates an edge uv only if u is one of the kNN of v and v is one of the kNN of u. This method produces a variable number of neighbors for each pixel, based on their local (in the spectral space) groupings. This method puts an emphasis on mutual neighbors and therefore clusters of similar pixels.

One problem with this method is that outliers often remain disconnected from the graph. Additionally, the k selected has an impact on the size of clusters that are represented as background. A large k can falsely treat many small clusters as part of the same background and a small k can leave small groups (but too big to be anomalies) disconnected from the graph. To maintain connectivity a minimum number of neighbors (which ignores the mutual neighbor requirement) can be implemented.

2.4 Density weighted k-nearest neighbors

For the proposed density weighted approach, the goal is to minimize the impact of pixels outside clusters. Each pixel is assigned a co-density score using k_{max} nearest neighbors. The co-density score is proportional to the sum of the distances to k_{max} neighbors divided by k_{max}. Threshold conditions are then applied to the distribution of co-density to provide several different values of k for the graph creation. Those pixels with the lowest density have few neighbors, while those with the highest use closer to k_{max} neighbors. From practical testing it appears that the threshold values, k_δ, should increase non-linearly. Pixels with lowest density should have few neighbors, those with

average density should be near $\frac{k_{max}}{2}$, and those with high density should be nearly k_{max}.

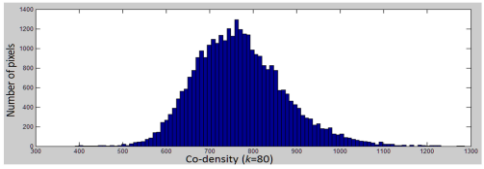

Fig. 4. A typical distribution of co-density. Although usually very Gaussian, co-density distributions can take other forms depending on scene content.

The best graph creation results in practice were obtained with six different k values based on the distribution of co-density. Because it is generally well represented by a normal distribution, the six density thresholds were selected based on the standard z-scores of the co-density histogram. The lowest of the size thresholds was z-score -2, followed by -1, 0, 1 and 2.

3. EXPERIMENTAL RESULTS

To test the four methods described above (kNN, locally weighted kNN, density weighted kNN, and mutual kNN) an experiment is performed utilizing two different graph segmentation techniques. Each edge creation method will be tested with normalized cuts segmentation [5] and modularity based segmentation [2, 3]. Both n-cuts and modularity clustering work on the same recursive splitting principle. For n-cuts, the graph is split based on the eigenvector associated with the second smallest eigenvalue of the $Laplacian\ matrix$; for modularity the largest eigenvector of the $modularity\ matrix$ is used [10].

Fig. 5. The data used for analysis comes from the HYDICE forest radiance scene. For this comparison a small 240×130 pixel tile containing a variety of features was selected.

Because each of these methods splits recursively from one large group to many small groups, the first split is of critical importance and any error there is maintained throughout the clustering. To test the various edge creation techniques, the first split alone will be analyzed. In addition to a variety of visual comparisons of the two class cluster map, the "optimal" first split was determined from manual analysis of the data structure for error comparison. The methods of modularity and n-cuts are not under examination here, instead only the methods used to create the graphs used in each technique are compared.

The data used to compare the methods are from the well-characterized HYDICE Forest Radiance scene [15]. The section of the image used for this analysis is a small region containing several man-made and natural features. The relatively small and simple scene was selected to make manual classification of specific features possible using traditional methods, specifically the Gaussian maximum likelihood (GML). The manually identified features provide "truth" data to compare the graph creation techniques.

3.1 Visual results

The first graph created utilized the simple kNN method with 60 neighbors for each pixel identified as edges. The number of neighbors chosen was a compromise between quality (many neighbors) and speed (few neighbors). This methodology does not account for the length of the edges in any way and places excess weight on outliers in the data.

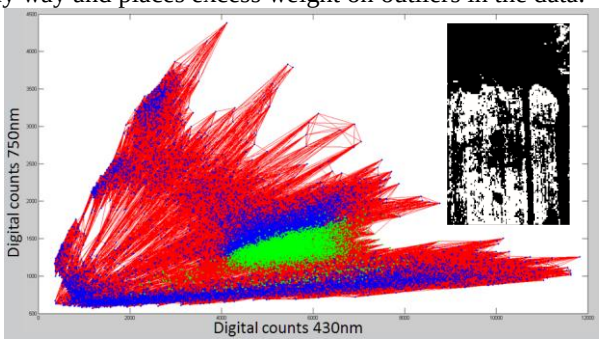

Fig. 6. The graph and first modularity split with cluster map overlay for the simple kNN method with k=60. Blue nodes are black pixels.

In **Figure 6**, the plot and the accompanying cluster map indicate that this method produces a "noisy" split, with many errors. From the plot of the graph, it is easy to see the over connectivity of the pixels that do not seem to belong to large groups. Although the graph is only plotted in two dimensions (750nm vs. 430nm), it is easy to see that some anomalous pixels are too well connected to the clusters with many long edges. This over connectivity of anomalies negatively impacts clustering. However, the results that follow show that the clustering improves with other techniques for edge creation. Although not obvious in the compressed few (in a spatial and spectral sense), the graph from the density weighted method in F**igure 7** shows far fewer over-connected nodes and total edges used.

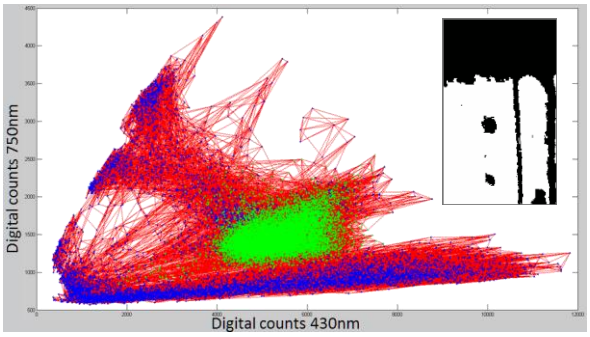

Fig. 7. This plot of the graph using the density weighted kNN technique ($k_{max} = 50$) can be contrasted with the simple kNN method shown in **Figure 6.**

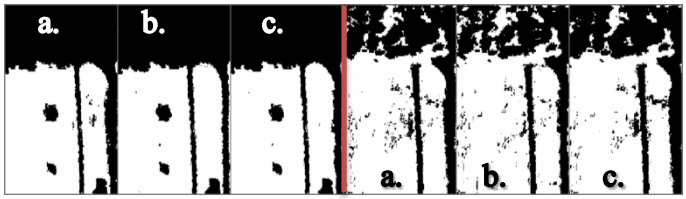

Fig. 8. Three 2-class cluster maps for modularity (right) and n-cuts (left) using: **a.** local ($k = 20, k_l = 10$), **b.** mutual ($k_{max} = 50$), and **c.** density kNN $\left(k_{max} = 50,\ k_\delta = 3, \frac{1}{10}k, \frac{1}{3}k, \frac{3}{4}k, \frac{9}{10}k, k\right)$.

Figure 8 shows the cluster map result for the locally weighted, mutual, and density weighted kNN methods from left to right for both modularity and n-cuts. The segmentation clearly improves compared to the simple kNN type and allows the use of fewer total neighbors. To better visualize the result compared to the manual "truth" map, error images are created using exclusive image differences. The "truth" was defined based on selecting training data for GML by visually examining the features found with modularity, therefore the shadows region and the tents which are part of the other group for the first n-cuts split appear to increase the error incorrectly.

Fig. 9. Here the error images are shown for the simple and density weighted methods for modularity and n-cuts. From left to right: original image, GML derived "truth" map, error image modularity simple kNN, error modularity density kNN, error n-cuts simple, error n-cuts density (error pixels are white).

It is clear that compared to simple kNN, the more advanced methods for graph construction (specifically the density weighted kNN) introduced here drastically improve clustering techniques for hyperspectral imagery. These results also apply to multispectral imagery, where graph based techniques can offer even greater improvement over traditional methods. An additional impact of the advanced methods is the decrease in total neighbors searched and in edges used (*i.e.* less computing resources).

4. CONCLUSIONS

The success of graph theory metrics and methods applied to imagery are dependent on the successful representation of an image as a graph. Turning an image into a graph is not a straightforward endeavor. Careful consideration must be taken when deciding which type of graph will work best.

Especially for clustering, it is clear that a graph representation that puts emphasis on representing the inherent community structure is desirable. The density weighted technique for generating a graph appears to put the most emphasis on the data's underlying community structure. Conversely, anomaly detection and potentially target detection methods may require less emphasis on community structure and number of connections and more emphasis on the weight of the connections. The methods described in this paper all assume that an edge is binary, but clearly the distance or similarity between the end points can be applied to create weighted graphs. While similar, weighted graphs may have different properties and exhibit different phenomenologies for clustering.

5. REFERENCES

[1] B. Basener, E. Ientilucci, and D.W. Messinger, "Anomaly detection using topology," in Algorithms and Technologies for Multispectral, Hyperspectral, and Uultraspectral Imagery XIII, S. Shen, Ed. SPIE, vol. 6565, April 2007

[2] R. Mercovich, T. Harkin, D.W. Messinger, "Utilizing the graph modularity to blind cluster multispectral satellite imagery," in 2010 WNYIPW, IEEE, November 2010.

[3] R. Mercovich, A. Harkin, D. W. Messinger, and B. Basener, "Automatic clustering of multispectral imagery by maximization of the graph modularity," in Algorithms and Technologies for Multispectral, Hyperspectral, and Uultraspectral Imagery XVII, S. Shen, Ed. SPIE, vol. 8048, April 2011.

[4] J. Albano, D. W. Messinger, A. Schlamm, and B. Basener, "Graph theoretic metrics for spectral imagery with application to change detection," in Algorithms and Technologies for Multispectral, Hyperspectral, and Uultraspectral Imagery XVII, S. Shen, Ed. SPIE, vol. 8048, April 2011.

[5] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation,' IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, August 2000.

[6] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Graph-Based Image Segmentation," Inter. Journal of Computer Vision, vol. 59, no. 2, 2004.

[7] M. Belkin and P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation," Neural computation, Vol. 15, No. 6, pp. 1373=1396, March 2006.

[8] A. Pothen, H. D. Simon, and L. Kan-Pu, "Partitioning sparse matrices with eigenvectors of graphs," SIAM Journal on Matrix Analysis and Applications, vol. 11 issue 3, Jul. 1990.

[9] Stanley Wasserman, Katherine Faust. Social Network Analysis: Methods and Applications, Cambridge University Press, 1994.

[10] M. E. J. Newman, "Modularity and community structure in networks," PNAS, Vol. 103, pp. 8577-82, June, 2006

[11] Mark Newman, Albert-Laszlo Barabasi, Duncan J. Watts, The Structure and Dynamics of Networks, Princeton Univ. Press, 2006.

[12] M. T. Gastner and M. E. J. Newman, "The spatial structure of networks," Eur. Phys. J. B, Vol. 49, 2006.

[13] A. Ziemann, D. W. Messinger, B. Basener, A. Schlamm, "Iterative convex hull volume estimation in hyperspectral imagery for change detection," in Algorithms and Technologies for Multispectral, Hyperspectral, and Uultraspectral Imagery XVI, S. Shen, Ed. SPIE vol. 7695, April 2010.

[14] Merwith, C., Parlitz, U., and Lauterbom, W., "Fast nearest-neighbor searching for nonlinear signal processing," Phys. Rev. E, vol. 62, pp. 2089-97, 2000.

[15] Manolakis, D., Marden, D., and Shaw, G.A., "Hyperspectral image processing for automatic target detection applications," MIT Lincoln Laboratory Journal, vol. 14, no. 1, pp. 79–116, 2003