NORMALIZED CUTS FOR SINGLE TREE ISOLATION FROM LIDAR

A Thesis submitted to the faculty of
San Francisco State University
In partial fulfillment of
the requirements for
the Degree

Master of Science

In

Geographic Information Science

by

Elias David Waggoner

San Francisco, California

May 2015

CERTIFICATION OF APPROVAL

I certify that I have read Normalized Cuts for Single Tree Isolation from LiDAR by Elias David Waggoner, and that in my opinion this work meets the criteria for approving a thesis submitted in partial fulfillment of the requirement for the degree Master of Science in Geographic Information Science at San Francisco State University.

Ellen Hines, Ph.D.
Professor

Leonhard Blesius, Ph.D.
Associate Professor

Bill Kruse,
Committee Member

NORMALIZED CUTS FOR SINGLE TREE ISOLATION FROM LIDAR

Elias David Waggoner
San Francisco, California
2015

This study demonstrates a developing method for the isolation of individual trees in a canopy from 3-dimensional laser scanning data. This method is capable of delineating single trees from a storied canopy, and is effective for individual tree isolation. This research compared the Normalized Cut (NCuts) method of single tree segmentation to a more commonly applied Hierarchical Watershed Transform (HWT) within an urban setting. For this study two field sites located within the urban build-up of San Francisco, CA were selected. The results of this work demonstrated an implementation of the Normalized Cut method coded using freeware libraries, and showed significant differences between the two methods, especially in storied canopies. This study also demonstrated the effectiveness and feasibility of using freeware coding tools for research, thereby enabling a wider range of research possibilities without the expense of large software packages.

I certify that the Abstract is a correct representation of the content of this thesis.

_____ _____

Chair, Thesis Committee Date

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figures Page

Introduction

With the use of airborne laser sensors, a large array of detailed information can be collected from vegetated areas without excessive field work, which usually comes at a high cost and often involves destructive sampling (Popescu, 2007). LiDAR (Light Detection and Ranging) is a technology that presents the ability to collect high density 3-D point cloud data within densely vegetated study areas. This can be done through hand editing or through commonly applied automated techniques (Gatziolis and Anderson, 2008; Gougeon, 1999; Zhao and Popescu, 2007; Kwak et al., 2007; Chen et al., 2006; Reitberger et al., 2007). One field that greatly benefits from these remote sensing technologies is forestry, where these data can be classified to simplify forest inventory and stand management through the creation of more detailed stand maps or individual tree identification (Popescu, 2007).

LiDAR system collects multiple returns per laser pulse as well as intensity of the return, while maintaining high levels of accuracy both horizontally and vertically (Popescu et al., 2007). However, there is still no single approach capable of returning optimal individual tree delineation in all cases. There are several available methods that attempt to delineate individual trees, each with advantages and limitations. This paper focuses on a comparison of two methods: the more traditional Hierarchical Watershed Transform (HWT) and the emerging method Normalized Cuts (NCuts).

In my study there are two goals: 1) to compare HWT to NCuts to demonstrate the strengths of NCuts, especially in multi-storied canopies and 2) to demonstrate the feasibility of coding with freeware libraries for this research application.

Developed by Shi and Malik (2000), the Normalized Cut is an unsupervised technique for image segmentation, meaning the cut is performed without user input of training information. This method approaches the tree delineation as a graph-partitioning problem, based around a set of global criteria (Carballido-Gamio et al. 2004). The basis is the separation of a region of interest (ROI) into a voxel space, or a set of 3-D volumetric pixels, which can be represented as a graph. As the distance between voxels increases, the similarities between them decreases and will drop to zero after a threshold. The final goal is to split the graph into segments based on maximized similarities between segment members, or nodes, while minimizing the similarity of the edges that connect the nodes (Reitberger et al., 2007).

My research compares the Normalized Cut method of single tree segmentation to a Hierarchical Watershed Transform within an urban setting. The results will demonstrate an implementation of the Normalized Cut method through the freeware Point Cloud Library (PCL - http://pointclouds.org/). My research question explore the difference in accuracy for Normalized Cuts single tree extraction versus that of the Watershed Transform. I will also demonstrate the effectiveness and feasibility of using freeware coding tools for research to enable a wider range of research possibilities without the expense of large software packages.

Methods

Study Area

For this study we used two field sites located within the city of San Francisco, CA (37.7833° N, -122.4167° W). Both sites are surrounded by urban build-up, but are part of

the largest green-spaces in San Francisco. The first is the panhandle of Golden Gate Park (37.7723° N, -122.4474° W) (Figure 1a), which stretches approximately 1.2 km west to east from the eastern boundary of the main area of Golden Gate Park into the middle of the city. This section of the Park has high pedestrian and vehicular traffic, and has multiple streets that cross through it. The Park is home to many different species of trees, with some of the most common being the blue gum eucalyptus (*Eucalyptus globulus*), Monterey pines (*Pinus radiata*), coast redwoods (*Sequoia sempervirens*) and the Monterey cypress (*Cupressus macrocarpa*). The second site is John McLaren Park (37.7203° N, -122.4184° W) (Figure 1b), located in south San Francisco, which is the second largest park in San Francisco, covering 312 acres. The dominant tree species in this park is the blue gum eucalyptus, which was introduced early in the development of the Park.

LiDAR Collection and Processing

The LiDAR data being used for this study were collected in the spring of 2010 through the American Recovery and Reinvestment Act (ARRA) Golden Gate LiDAR Project (Hines et al. 2010). These data were classified to create a 0.5 m canopy height model (CHM) by subtracting ground elevation values in a digital elevation model (DEM) from the elevation values of a digital surface model (DSM). A raster map layer was then returned where the z- values stored for each point represented a height above ground, instead of an elevation (Loos 2009, Naesset et al. 2002, Gougeon 1995). These processes were performed using the LP360 software package developed by QCoherent and the Lastools suite of tools developed by Martin Isenberg.

Hierarchical Watershed Transform

A Watershed Transform can be thought of as a flooding simulation, where the canopy height model (CHM) is inverted. This process creates an image where tree tops are represented by low points, or minima, which resemble catchment basins. The edge-detection algorithm portion of the watershed delineation then creates a boundary where a basin would overflow if filled with water, to prevent two basins from merging (Vincent and Soille, 1991). These boundaries are called watershed lines and are used to delineate the individual trees or tree stands. This study implemented a more advanced version of this method, called the Hierarchical Watershed Transform (HWT), which generates a set of nested partitions at several resolution levels.

For a HWT a gradient image is used instead of the original. When using LiDAR data, the intensity of the return pulse can be used to create the gradient image (Kwak et al., 2007). The version of HWT we used calculates the boundaries with either a pixel or region level precision (Figure 2). Pixel level precision indicates that the image is viewed as a weighted graph, where the intensity values of the pixels are the primary unit. In contrast, the region level precision indicates that the image, when viewed as a weighted graph, has vertices representing the primitive catchment basins (PCB), or largest and simplest segmented regions, of each regional minimum.

The HWT is formulated as a graph optimization problem under the framework of the Image Foresting Transform (IFT) (Falcao et al., 2004). This means that the data are converted into a weighted graph represented by $G=(V,E,w)$, where there exists two sets V and E, representing the vertices and edges, respectively. These two sets are weighted by a

cost function, shown here as *w*. This structure results in partitions by labeling the different digitized edges, with the labels corresponding to separate catchment basins. Some of these edges can then be suppressed, to create larger watershed regions. The edge weights determine which edges are suppressed, with the finest partition suppressing all edges and corresponding to the classical watershed.

To derive the hierarchies between the most extreme segmentation levels, synchronous flooding was used so all parts of the image were submerged at a constant speed. This results in the assignment of different weights to the edges of the weighted graph. These calculations were performed by ranking regions by the volume criterion, which is the combined weight of image contrast by region and the region's size.

A well-documented problem with the basic watershed transform is over-segmentation of the image where the canopy of a single tree is split into multiple objects (Roerdink & Meijster, 2001; Popescu, 2003; Zhao and Popescu, 2007). This problem has largely been solved through the inclusion of marker-controls to aid in identifying individual tree tops (Zhao and Popescu, 2007; Kwak et al., 2007; Chen et al., 2006). Marker controls were implemented in this study to limit over-segmentation, chosen from GPS data collected and corrected in the lab for a standard error of approximately one meter.

Normalized Cuts

The method compared to the HWT was the Normalized Cut method introduced by Shi and Malik (2000) (see Figure 3 for a flow chart of Ncut methods). The basic approach of Normalized Cut treats the segmentation of images as a graph partitioning

problem, where the edges joining two sets of graphs are removed, and the total weight of those removed edges are calculated as the dissimilarity between the graphs; this dissimilarity is called the cut. Typically this cut has been minimized to create the optimum partitioning, however this favors very small nodes and graphs, thereby creating extremely small disjointed cuts in larger datasets (Shi and Malik, 2000; Wu and Leahy, 1993).

The basis of this method, when applied to LiDAR, is the separation of a ROI into a 3-dimensional volumetric pixel (voxel) space, which can be represented as a graph G={V,E}, where V is the set of voxels representing the segmentation nodes, and E the set representing the edges between nodes. The key idea is that as the distance between voxels increases, the similarities between them decreases and will eventually drop to zero. The cut is implemented on the sparse graph using similarities larger than empirical threshold. The sparse graph enables the near linear, versus exponential growth of computation time. The algorithm splits the graph into segments by maximizing the similarities between the individual voxels, while minimizing the similarity of the different graph segments. The resulting cost function is presented in equation (1), with equations (2) and (3) as inputs.

$$NCut(A,B) = \frac{Cut(A,B)}{Assoc(A,V)} + \frac{Cut(A,B)}{Assoc(B,V)} \quad (1)$$

$$Cut(A,B) = \sum_{i \in A, j \in B} w_{ij} \quad (2)$$

$$Assoc(A,V) = \sum_{i \in A, j \in V} w_{ij} \quad (3)$$

Equation (4) represents the sum of weights between two graph segments, A and B, and equation (5) is the sum of the weighted edges ending at segment A. In both equations, he value w_{ij} represents the weight value between two points, *i* and *j*.

A generalized eigenvalue problem, represented by equation (4), then solves the minimization cut of NCut(A,B) (Reitberger et al., 2007).

$$(D - W)y = \lambda Dy \tag{4}$$

In this equation the *n x n* weighting matrix *W* represents the weights between all voxels within a graph. The eigenvector is represented by *y* and the eigenvalue by λ. This equation is derived from the Rayleigh quotient, which can be minimized by the smallest eigenvector of the matrix (in this case the matrix is represented by *D - W*), and the minimum value is the corresponding eigenvalue. This means that, because the first smallest eigenvalue of the matrix is 0, the second smallest eigenvalue is the solution for NCuts, and the corresponding eigenvector determines how to partition the graph (de Carvalho et al., 2010). To calculate the weights for the matrix *W* equation (5) is used.

$$w(i,j) = e^{-F(i,j)} * e^{-X(i,j)} * e^{-Z(i,j)} \tag{5}$$

In this equation, *F(i,j)*, *X(i,j)*, and *Z(i,j)* are weighted values for the intensity, horizontal data, and vertical data respectively. These values are more thoroughly explained in

equations (6) – (8) respectively. This matrix is a variation on a simple adjacency matrix, where instead of a binary matrix measuring the relationship between each point, the weight between two vertices is used instead. A threshold is also implemented below which all calculated values become zero, to reduce computational complexity. The next matrix, *D,* is derived from *W* and is the number of edges incident to the vertex. In a two dimensional image this is computationally similar to performing a *k-nearest neighbor* search, or radius search, to find the number of connected nodes (Zhao and Liu, 2007). However, in a LiDAR dataset, the data is unstructured, meaning that the number of connected nodes is variable. In this instance, all other points within the graph must be initially considered when calculating the number of nodes, and then limited by a "no impact" threshold.

Therefore, the minimum solution for y_1 represents the second smallest eigenvalue, and may only have two indicator values (+1,-1). These indicator values must then be discretized using a variable voxel threshold based upon the study site. For our study, the variable was set to 40 voxels based on the average number of points and collected tree heights.

NCuts was implemented using C++ and pre-existing libraries available from the Point Cloud Library project, which is a suite of coding libraries designed for 2D and 3D image and point cloud processing (http://www.pointclouds.org/). The initial point cloud data were left in an ASCII format in a text file containing X, Y, Z locations and the intensity value of each point. These points were read into an array, which was used to

create the *n x n* weighting matrix *W*, based on equation (5) with equations (6) through (8) as inputs.

$$F(i,j) = \left(\frac{|f_i - f_j|}{\sigma_z}\right)^2 \; where \; f_i = \{I_{mean}, W_{mean}\} \tag{6}$$

In this equation I_{mean} and W_{mean} represent the average mean intensity of a voxel and the pulse width of the waveform. For this study the pulse width is not available, due to the use of discrete return LiDAR, however the algorithm can run without it, cutting the data based on spatial coordinates and pulse intensity.

$$X(i,j) = \left(\frac{D_{ij}^{XY}}{\sigma_{xy}}\right)^2 \tag{7}$$

$$Z(i,j) = \left(\frac{D_{ij}^{Z}}{\sigma_z}\right) \tag{8}$$

The next two components, from equations (7) and (8), are X(i,j) and Z(i,j). These weights are used to control the distances between the voxels using the quadratic Euclidean distance between the voxels, with D_{ij}^{XY} representing the horizontal and D_{ij}^{Z} the representing the vertical. These functions calculate the most important impact factors of the similarity, dependent upon the distance between voxels, where F(i,j) is the quadratic Euclidian distance between two features, derived from the reflection values of the voxels, represented in equation (6).

The resulting matrix, *W*, was then transferred to a triplet format, which is a small structure designed to hold the location and value of a non-zero data point. This triplet format is used to smoothly transfer data into a sparse matrix through tools from the freeware coding library Eigen, which is bundled with PCL. Based on our field data, we used the empirical weighting values $\sigma_f = 0.5$, $\sigma_{xy} = 8.47$ m, $\sigma_z = 14.12$ m to keep the impact factors of *F(i,j)*, *X(i,j)*, and *Z(i,j)* controlled. These values were chosen from the data and through a Monte-Carlo simulation to represent the proper weighting points for quadratic Euclidean distance between the voxels, the crown width and height of the trees, and the average intensity values respectively. The value of σ_z is kept larger than the value for σ_{xy} because typically tree heights are larger than their crown diameter.

The cut value for each voxel is based on the results of equation (4), which can be calculated in a number of ways. In my project I experimented with two different eigensolvers. The algorithms I used were, minimal residual solver (MINRES), and Lanczos with a QR transform.

The first eigensolver I implemented was a version of MINRES written for use within the Eigen libraries. This method is designed to approximate the exact solution of equation (4) by creating a vector within the Krylov subspace of the graph. This means that the equation works by multiplying vectors with the primary input matrix and then calculating the eigenvalues and eigenvectors based on the results. For the MINRES algorithm, this is used to minimize the Euclidean norm of the residual, producing the estimated eigenvalue.

The next algorithm was a direct implementation of the Lanczos algorithm with a QR transform, tested through Eigen, a linear algebra library called LAPACK, and an interpretative language called GNU Octave. The Lanczos algorithm is an adaptation of the power iteration method used to find the most useful eigenvalues and eigenvectors of a system, using a limited number of operations. The algorithm works by iteratively solving for potential normalized eigenvectors of a matrix, which will correspond to the largest eigenvalues. As the values converge, they correspond to the largest eigenvalue of the matrix, and the algorithm can then be restricted to find the second largest. This will output a tridiagonal matrix of values that requires another algorithm, such as QR, to transform the results into eigenvalues and eigenvectors.

The resultant data from both of these eigensolvers were used to create the cut threshold. Any values that fall below the second eigenvalue are separated into a new class if the voxel count for each class exceeds the maximum limit. This process splits the class G into two new class G_1 and G_2, creating areas of points for two distinct trees. This process is iterated until the total number of voxels reaches our threshold of 40. The output data consists of voxel values and locations, indicating which voxels fall within each of the split graphs, with the splits corresponding with the outer edges of each individual tree.

Field Data Collection

To perform the accuracy assessment *in situ* data were collected from plots located at each study site. In the Panhandle, data were collected from all of the trees located between the cross-streets of Cole and Clayton (Figure 4). Data were also collected from six circular plots, 30 m in diameter, located within John McLaren Park (Figure 5) to get a

representative sample of the variation in species and stand age in the area. At both sites tree height was measured using both a MDL LaserAce 300 Digital rangefinder as well as through the application of trigonometric principles based on distance from the tree, and the angle from the observer to the base and top of the tree. These data were collected using a meter tape and a *Suunto* clinometer.

Results

The results for each method were split into three tree layers based upon the height above ground. The first layer was anything less than 15 m tall, the second was between 15 m and 25 m, and the final layer was classified as anything above 25 m. For the lower layer of trees, the HWT method detected less than 5% of the trees that were identified in the field. In the intermediate layer the HWT found approximately 25% of the trees. In the uppermost layer, considered the canopy layer the HWT performed well, correctly identifying 85% of the trees within this layer. Overall the HWT method correctly identified less than 60% of the trees within both study sites.

All implementations of the Normalized Cut that I wrote for this study only accurately performed at the lowest height layer. At the lowest layer (<15 m) my NCuts implementation produced an average accuracy of 45%. At the middle and upper classification layer, my implementation failed to perform, resulting in an accuracy of 20%, and 10% respectively. The final output data all were all classified in horizontal bands across the study sites as seen in Figure 6.

Based on the handheld GPS data collected *in situ,* individual tree locations were found to be off by an average of one meter for measurements taken at John McLaren Park

and in the interior of the panhandle of Golden Gate Park. Measurements taken from the outer edges of the panhandle, along the sidewalk, were found to be off by as many as 10 meters. Tree heights collected *in situ* were found to differ from LiDAR measurements by an average of one meter, however the individual measurements were found to differ by as many as four meters.

Discussion

Overall the implementations of the Normalized Cuts method of individual tree identification written for this study failed to outperform the Hierarchical Watershed Transform at any of the higher canopy levels. The algorithm I wrote for this study consistently produced inaccurate results at all canopy layers, tending towards classifying the data in horizontal bands (Figure 6). At the lowest canopy level there is a significantly higher classification rate from NCuts, however the accuracy was appears to be the coincidental result of smaller trees being located within the horizontal classification band. At the middle and high classification heights, the accuracy dropped significantly due to the horizontal classification only capturing parts of a tree. These horizontal sections did consistently classify tree tops at each level, however the over-classification into other trees lowered the overall accuracy.

The inaccurate results of my implementation of NCuts were consistent through all eigensolver implementations, using Eigen libraries, Octave and LAPACK. In all of the methods attempted, the resulting cuts were either horizontal or scattered around the data (Figure 7). This would seem to indicate that errors are caused either in the calculation of the input matrices, or in the way the data are being input into the solvers. These are both

areas of my code that have had to be re-written multiple times through the duration of the project, as I became more comfortable with linear algebra.

I worked alongside Bill Kruse in an effort to find the source of my errors, based on a functional version of the NCuts algorithm he had written in the Python programming language. This work helped verify that my weighting factors were calculating correctly, however these efforts were not enough to elucidate the errors within my own algorithm.

The GPS errors made locating and comparing some of the trees difficult, and could be a source of error in the analysis. This can be attributed to a combination of the LiDAR data being collected four years prior to the fieldwork, as well as the introduction of human error in the *in situ* ground-truthing methods.

Conclusions

The goal of this research was to compare the more commonly applied Hierarchical Watershed Transform (HWT), and the Normalized Cuts (NCuts) method. Overall, the study failed to demonstrate a working version of the NCuts method and was unable to give an accurate comparison of the two algorithms. The failure to correctly implement these concepts led to very low accuracies and a horizontal classification scheme, as opposed to vertical. Although I was not able to demonstrate the efficacy of freeware software libraries, this error seems to be a flaw with my own ability to implement the tools available. I believe that future research studies could be performed without the expense of large software packages, however it would take a researcher much more familiar with programming. In the literature, it is suggested that by taking advantage of the 3-dimensional attributes of LiDAR data, more thorough tree isolations

can be performed. Future studies could work on implementing the code with other eigensolvers, and include an implementation GPU processing, or could implement the code with higher point density LiDAR data for more accurate delineations. These data could then also be used for more thorough stand management studies, such as tree species identification, canopy structure and biomass studies.

References

Carballido-Gamio, J., Belongie, S.J., and Majumdar, S. (2004). Normalized cuts in 3-D for spinal MRI segmentation. *IEEE Transactions on Medical Imaging*, 23 (1), pp. 36-44.

Chen, Q., Baldocchi, D., Gong, P., and Kelly, M. (2006). Isolating individual trees in a savanna woodland using small footprint LiDAR data. *Photogrammetric Engineering & Remote Sensing* 72 (8), pp. 923-932.

de Carvalho, M.A.G., Pinto, T.W., Ferreira, A.C.B., & Júnior, R.M.C., (2010). Image Segmentation Using Watershed and Normalized Cut.

Gatziolis D. and H. Andersen., 2008. A guide to LiDAR data acquisition and processing for the forest of the Pacific Northwest. United States Department of Agriculture, Forest Service. Portland, OR. 32p.

Gougeon, F.A., (1995). A crown-following approach to the automatic delineation of individual tree crowns in high spatial resolution digital images. *Canadian Journal of Remote Sensing* 21 (3), pp. 274-284.

Klava, B., and Hirata, N.S.T., (2009). Interactive Image Segmentation with Integrated Use of the Markers and the Hierarchical Watershed Approaches. *VISAPP, International COnference on Computer Vision Theory and Applications*, pp. 186-193.

Kwak, D.A., Lee, W.K., Lee, J.H., Biging, G.S., and Gong, P. (2007). Detection of individual trees and estimation of tree height using LiDAR data. *Journal of Forest Research*, 12, pp. 425-434.

Lehmann, Gaetan, (2005). Minima Imposition Image Filter. *The Insight Journal* Vol. 63, pp. 1-3.

Loos, R. (2003). Understanding the forest structure: Development of tools for identification and delineation of individual trees using LiDAR (Master's thesis). Retrieved from https://dspace.library.uvic.ca:8443/handle/1828/3110

Naesset, E., & Okland, T. (2002). Estimating tree height and tree crown properties using airborne scanning laser in a boreal nature reserve. *Remote Sensing of Environment* 79, pp. 105-115.

Papon, J., Abramov, A., Schoeler, M., and Worgotter, F., (2013). Voxel Cloud Connectivity Segmentation – Supervoxels for Point Clouds. *IEEE Conference on Computer Vision and Pattern Recognition.* Pp. 2027-2034.

Pilger, N., (2008). Coupling LiDAR and high-resolution digital imagery for biomass estimation in mixed-wood forest environments. *ASPRS 2008 Annual Conference Proceedings*.

Popescu, S.C., Wynne, R.H., Nelson, R.F. (2003). Measuring individual tree crown diameter with LiDAR and assessing its influence on estimating forest volume and biomass. *Canadian Journal of Remote Sensing* 29(5), pp 564-577.

Popescu, S.C., Wynne, R.H., and Scrivani, J.A. (2004). Fusion of small-footprint LiDAR and multispectral data to estimate plot-level volume and biomass in deciduous and pine forests in Virginia, USA. *Forest Science* 50(4), pp 551-565.

Popescu, S.C. (2007). Estimating biomass of individual pine trees using airborne LiDAR. *Biomass and Bioenergy* 31, pp. 646-655.

Reitberger, J., Heurich, M., Krzystek, P., and Stilla, U., (2007). Single tree detection in forest areas with high-density LiDAR data. *Photogrammetric Image Analysis*, (Sept.) pp. 19-21.

Roerdink, J.B.T.M., & Meijster, A., (2001). The watershed transform: Definitions, algorithms and parallelization strategies. *Fundamenta Informaticae*, 41, pp. 187-228.

Shi, J., and Malik, J.m (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 22, pp. 888-905.

Vincent, L. & Soille, P. (1991). Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*13 (6), pp. 583-598.

Zhao, K., and Popescu, S., (2007). Hierarchical watershed segmentation of canopy height model for multi-scale forest inventory. *IAPRS,* Vol. XXXVI, Part 3, pp 463-441.

Zhao, K., and Popescu, S., (2009). LiDAR-based mapping of leaf area index and its use for validating GLOBCARBON satellite LAI product in a temperate forest of the southern USA. *Remote Sensing of Environment*, 113, pp. 1628-1645.

Figures

Figure 1. In the upper left (a) shows the Panhandle of Golden Gate Park, and (b) John McLaren Park, both located within San Francisco.

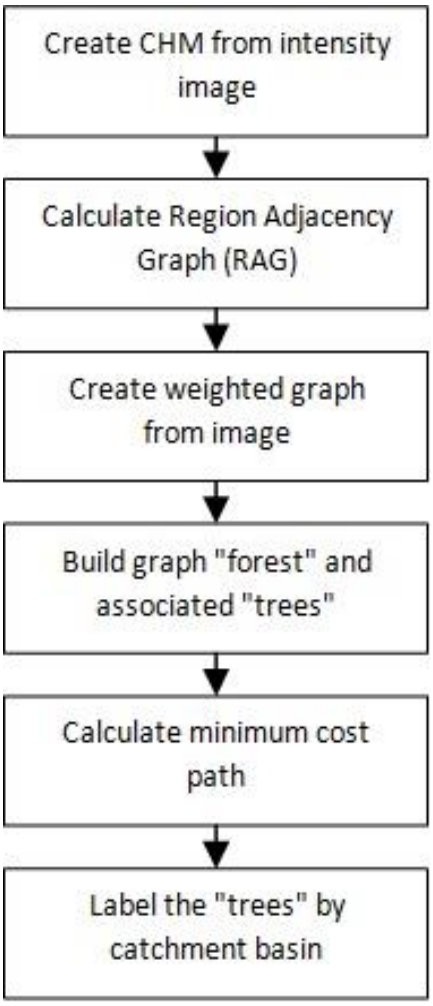

Figure 2. Flowchart of Hierarchical Watershed Transform methods

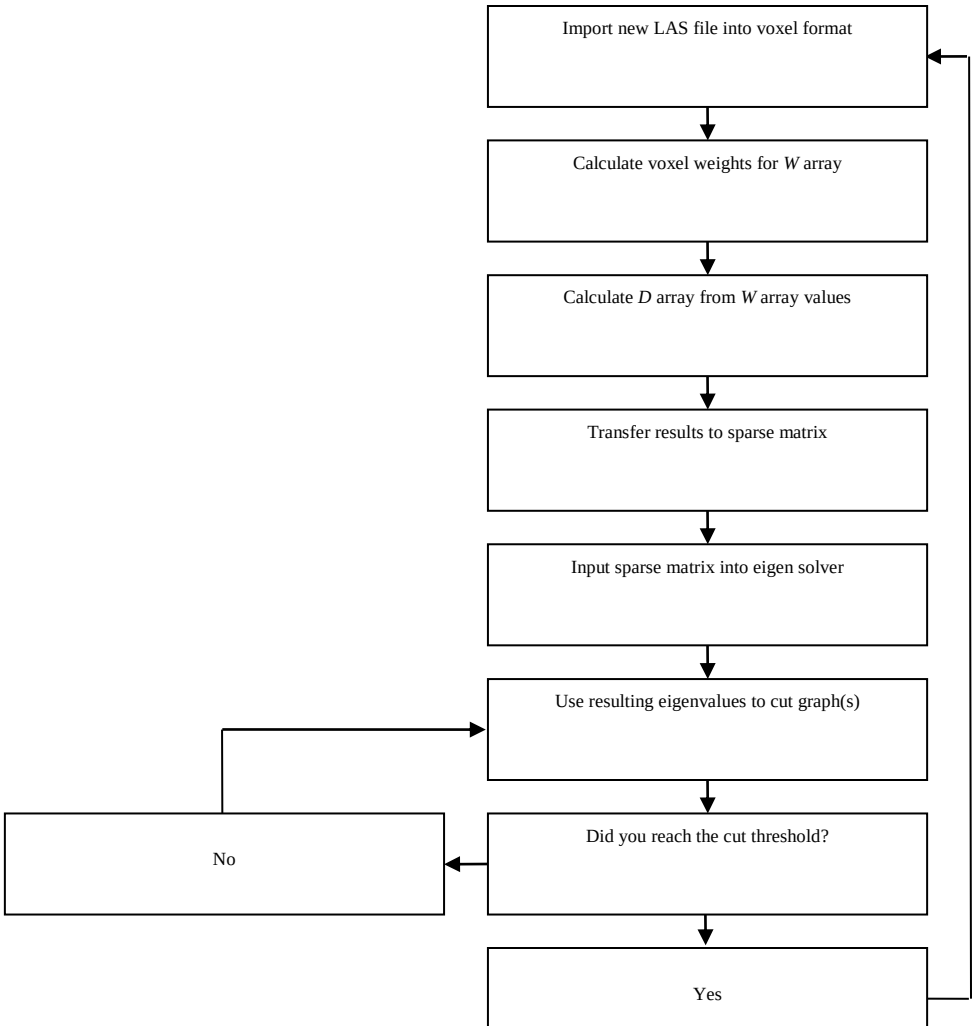

Figure 3. Flowchart of Normalized Cuts Methods

Figure 4. Outline of sample sites in Golden Gate Park Panhandle

Figure 5. Locations of sample sites from John McLaren Park

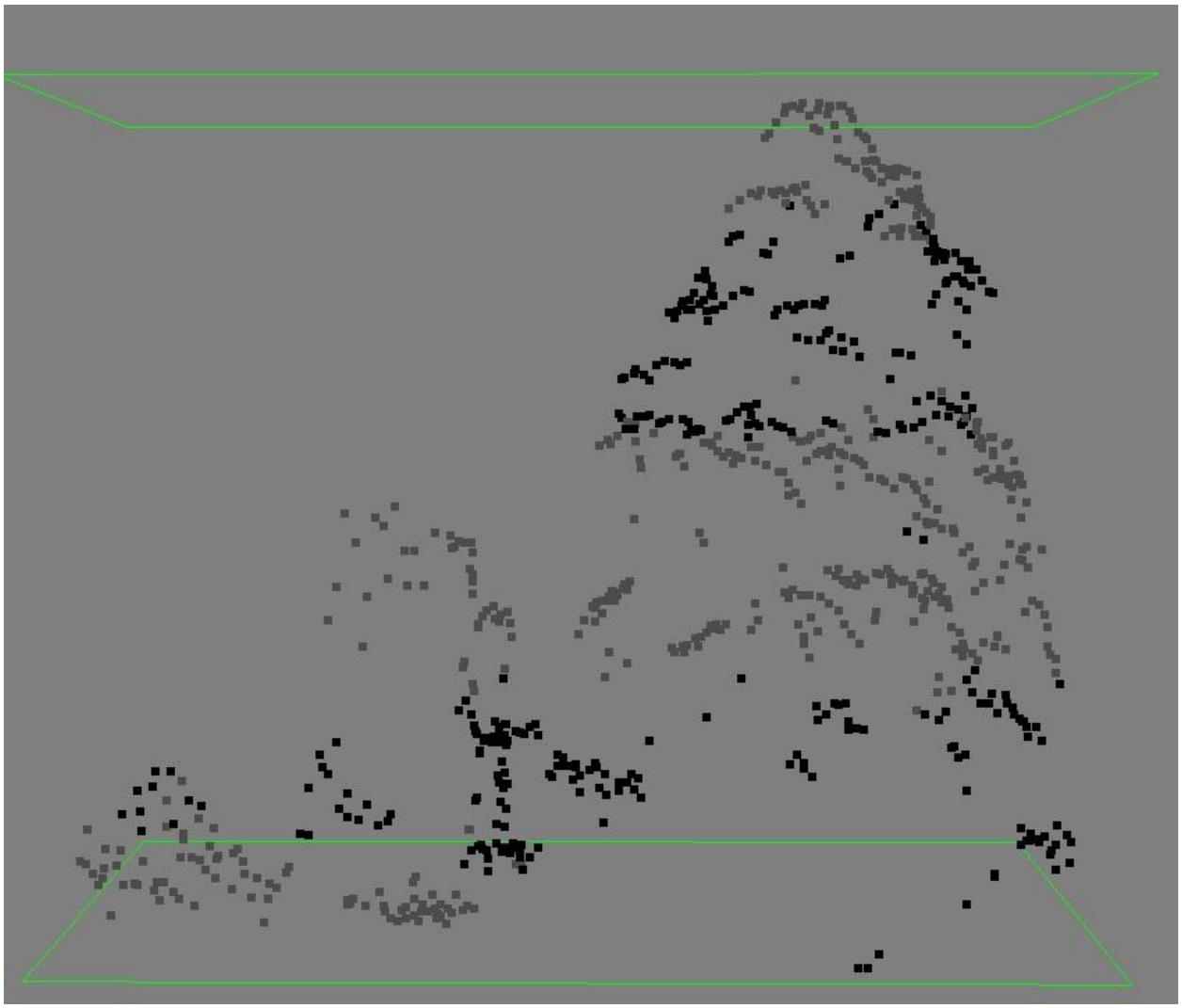

Figure 6. Horizontal banding in the Normalized Cuts results. Two classes shown for three tree tops. Class 1 - Grey and Class 2 - Black